

BAB 1

PYTHON

1.1 Sejarah Python

Python merupakan bahasa pemrograman yang sangat populer. Guido van Rossum merupakan yang membuat bahasa 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 disebarakan 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. Bahasa 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 untuk 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 - perbedaan dari yang telah ditemukan dan diketahui sebelumnya. Berikut adalah contohnya:

1. Print

Pada Python2, print diperlakukan seperti statemen ketimbang sebuah function.
`print "Esi Vidia Rahmadani"`

Sedangkan pada Python 3, print diperlakukan sebagai function.

`print ("Esi Vidia Rahmadani")`

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 diisi 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

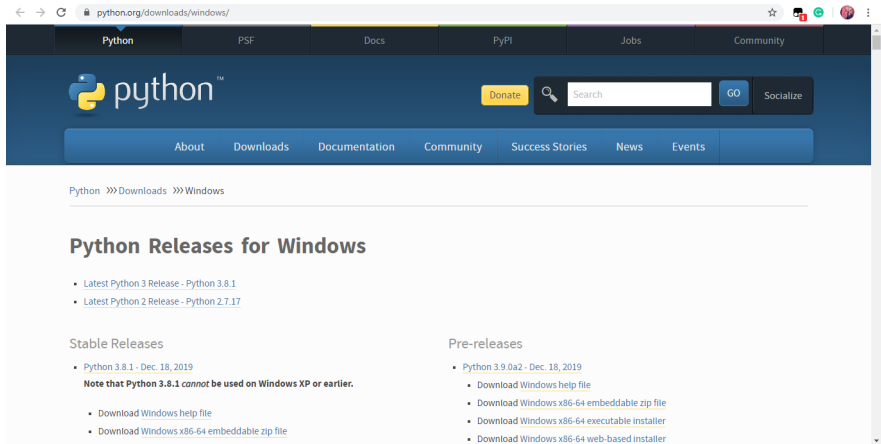
Selain perbedaan mendasar antara Python2 dan Python3 secara sintatikal, pada fakta lainnya adalah Python 2.7 akan dihentikan dukungannya per tahun 2020 dan Python3 akan terus dikembangkan dengan fitur yang lebih banyak lagi kedepannya, dan tentu saja yang lebih penting adalah perbaikan dari bug dan performa. 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

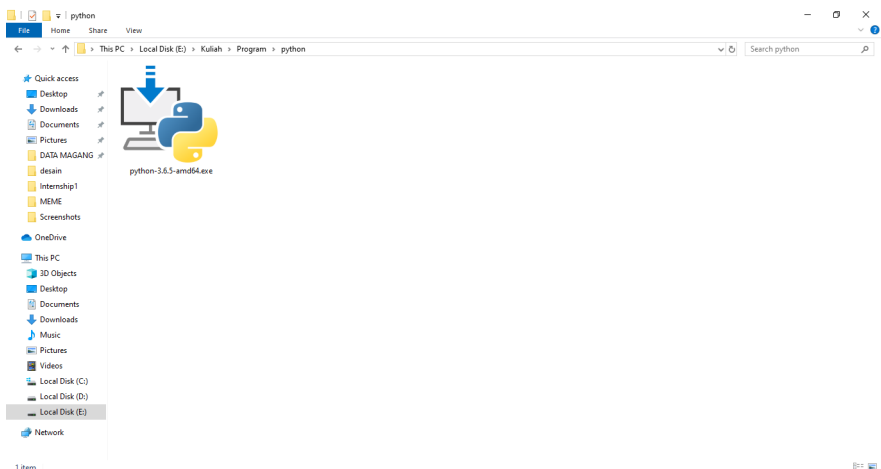
Instalasi python pada Windows sangatlah mudah. Langkah-langaknya yaitu sama seperti menginstall software Windows yang lainnya, next-next dan finish. Tetapi, terdapat konfigurasi yang harus dipilih terlebih dahulu ditengah-tengah proses instalasi, agar perintah pada Python dapat dikenali di CMD. Python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



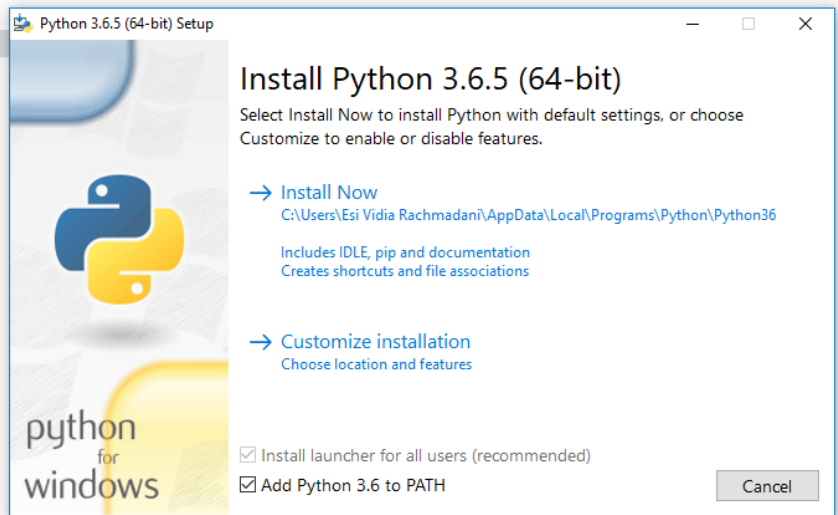
Gambar 1.2 Website python.org

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.



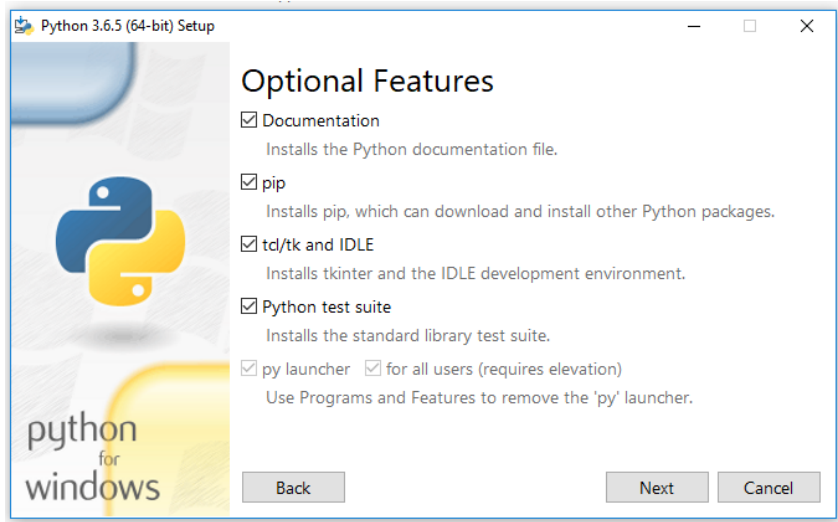
Gambar 1.3 File Python

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



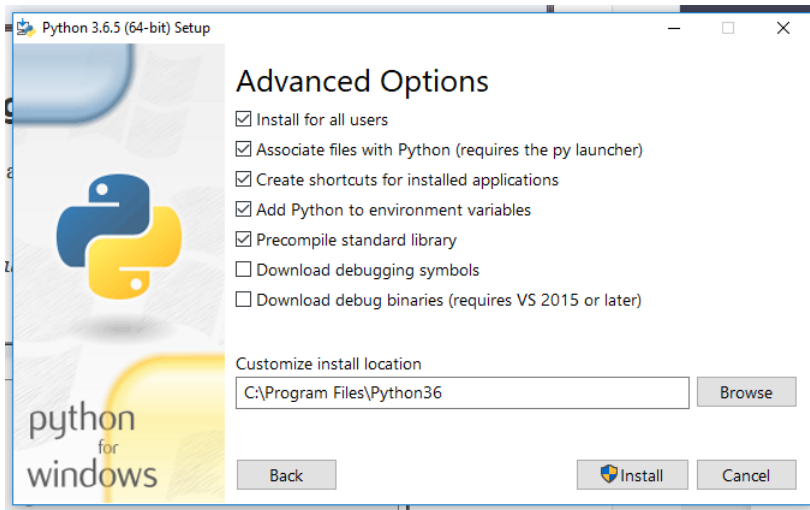
Gambar 1.4 Customize Installation

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



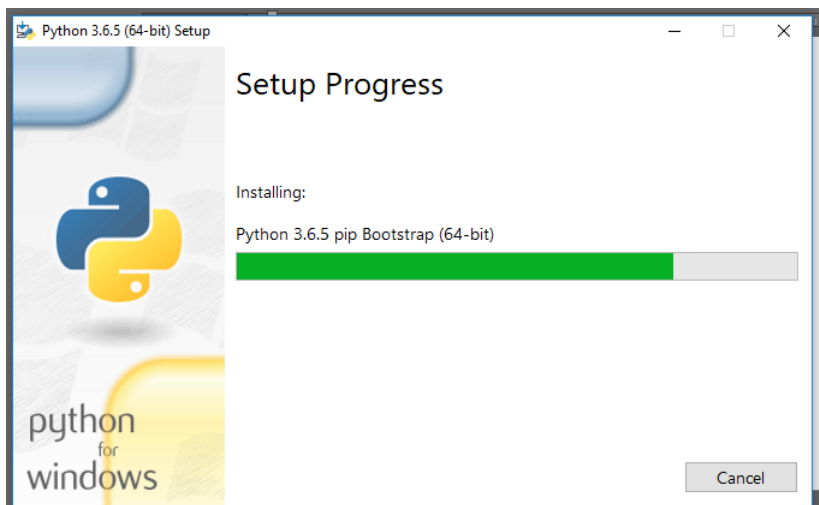
Gambar 1.5 Optional Features

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



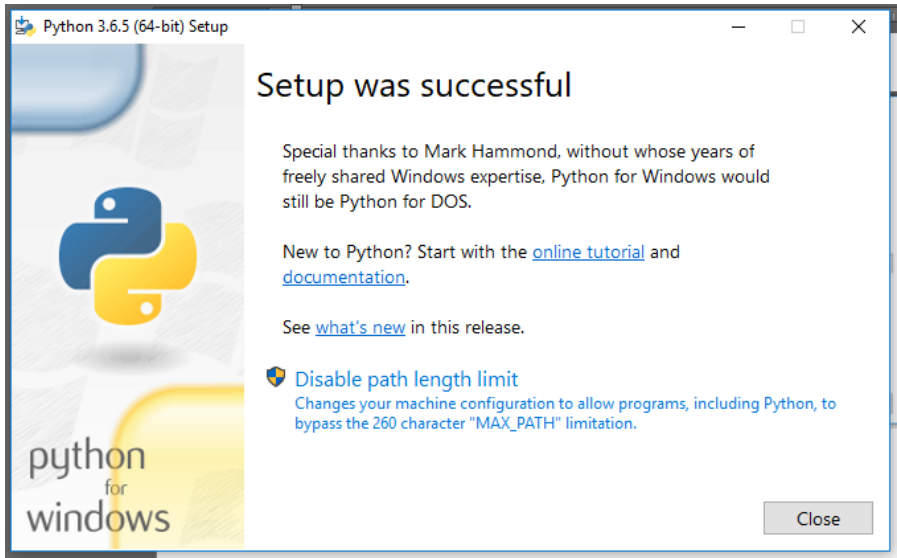
Gambar 1.6 Advanced Options

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 memasukannya 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 lainnya.

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:

```
1 #Ini
2 #Adalah
3 #Sebuah
4 #Komentar
5 print("Hai Dunia") #ini juga komentar
6 #print("Hai")
7 #komentar
8 #bisa berisi
9 #spesial karakter #$&*() ,!@./; ' [%^]\
10 #mencetak
11 #nama
12 print("Vidia")
13 #mencetak
14 #angka/ integer
15 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 dibandingkan dengan bahasa pemrograman yang lainnya. Berikut merupakan beberapa tipe data dari bahasa pemrograman Python:

Tabel 1.1 Tipe Data

Tipe Data	Contoh	Penjelasan
Float	7.13 atau 0.17	Menyatakan bilangan yang mempunyai koma
Complex	3 + 4j	Menyatakan pasangan angka real dan imajiner
String	"Semangat Kawan"	Menyatakan karakter/kalimat bisa berupa huruf angka, dll (diapit tanda " atau ')
Tuple	('abc', 836, 5.24)	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	['abc', 836, 5.24]	Data untaian yang menyimpan berbagai tipe data dan isinya bisa diubah-ubah
Dictionary	'nama': 'esi', 'id': 3	Data untaian yang menyimpan berbagai tipe data berupa pasangan penunjuk dan nilai

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

```

1 #tipe data Float
2 print(4.17)
3 #tipe data Complex
4 print(9e)
5 #tipe data String
6 print("Semangat Kawan")
7 print('Kita Pasti Bisa')
8 #tipe data Tuple
9 print((2,4,7,8,9))
10 print(("dua", "empat", "tujuh"))
11 #tipe data Boolean
12 print(False)
13 #tipe data Integer
14 print(98)
15 #tipe data Hexadecimal
16 print(7c)

```

```

17 #tipe data List
18 print([2,4,7,8,9])
19 print(["dua", "empat", "tujuh"])
20 #tipe data Dictionary
21 print({"Nama":"EsiVR", 'Umur':22})
22 #tipe data Dictionary dimasukan ke dalam variabel biodata
23 biodata = {"nama":"Andi", 'umur':21} #proses inisialisasi variabel
    biodata
24 print(biodata) #proses pencetakan variabel biodata yang berisi tipe
    data
25 Dictionary
26 type(biodata) #fungsi untuk mengecek jenis tipe data. akan tampil <
    class
27 '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 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 :

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 dimasukkan.

Contoh penggunaan variabel dalam bahasa pemrograman Python.

```

1 #proses memasukan data ke dalam variabel
2 nama = "Dava Evar"
3 #proses mencetak variabel
4 print(nama)
5 #nilai dan tipe data dalam variabel dapat diubah
6 umur = 17 #nilai awal
7 print(umur) #mencetak nilai umur
8 type(umur) #mengecek tipe data umur
9 umur = "tujuh belas" #nilai setelah diubah
10 print(umur) #mencetak nilai umur

```

```

11 type (umur) #mengecek tipe data umur
12 namaDepan = "Arif"
13 namaBelakang = "Budiman"
14 nama = namaDepan + " " + namaBelakang
15 umur = 29
16 hobi = "Traveling"
17 print("Biodata\n", nama, "\n", umur, "\n", hobi)
18 #contoh variabel lainya
19 inivariabel = "Haaaii"
20 ini_juga_variabel = "Hooo"
21 _inivariabeljuga = "Hi"
22 inivariabel222 = "Haaa"
23 panjang = 7
24 lebar = 9
25 luas = panjang * lebar
26 print (luas)

```

1.10 Operator

Operator adalah konstruksi yang dapan memanipulasi nilai dari operan. Sebagai contoh operasi $3 + 2 = 5$. Disini 3 dan 2 adalah operan dan + adalah operator. Bahasa pemrograman Python mendukung berbagai macam operator, diantaranya :

- Operator Aritmatika (Arithmetic Operators)
- Operator Perbandingan (Comparison (Relational) Operators)
- Operator Penugasan (Assignment Operators)
- Operator Logika (Logical Operators)
- Operator Bitwise (Bitwise Operators)
- Operator Keanggotaan (Membership Operators)
- Operator Identisas (Identity Operators)

Mari kita membahasnya satu-persatu.

1.10.1 Operator Aritmatika

Tabel 1.2 Tipe Data

Operator	Contoh	Penjelasan
Pengurangan -	$7 - 2 = 5$	Mengurangi nilai operan di sebelah kiri menggunakan operan di sebelah kanan
Penjumlahan +	$3 + 5 = 8$	Menjumlahkan nilai dari masing-masing operan atau bilangan
Pembagian /	$12 / 3 = 4$	Untuk membagi operan di sebelah kiri menggunakan operan di sebelah kanan
Perkalian *	$2 * 4 = 8$	Mengalikan operan/bilangan
Pangkat **	$8 ** 2 = 64$	Memangkatkan operan disebelah kiri operator dengan operan di sebelah kanan operator
Sisa Bagi %	$11 \% 2 = 1$	Mendapatkan sisa pembagian dari operan di sebelah kiri operator ketika dibagi oleh operan di sebelah kanan
Pembagian Bulat //	$10 // 3 = 3$	Sama seperti pembagian. Hanya saja angka dibelakang koma dihilangkan

Contoh penggunaan Operator Aritmatika dalam bahasa pemrograman Python:

```

1 #file /python_dasar/operator_aritmatika.py
2 #OPERATOR ARITMATIKA
3 #Penjumlahan
4 print(17 + 3)
5 apel = 6
6 jeruk = 8
7 buah = apel + jeruk #
8 print(buah)
9 #Pengurangan
10 hutang = 15000
11 bayar = 9000
12 sisaHutang = hutang - bayar
13 print("Sisa hutang Anda adalah ", sisaHutang)
14 #Perkalian
15 panjang = 4
16 lebar = 13
17 luas = panjang * lebar
18 print(luas)
19 #Pembagian
20 kue = 100
21 anak = 4
22 kuePerAnak = kue / anak
23 print("Setiap anak akan mendapatkan bagian kue sebanyak ", kuePerAnak
24 )
25 #Sisa Bagi / Modulus
    bilangan1 = 16

```

```

26 bilangan2 = 7
27 hasil = bilangan1 % bilangan2
28 print("Sisa bagi dari bilangan ", bilangan1, " dan ", bilangan2, "
    adalah ",
29 hasil)
30 #Pangkat
31 bilangan3 = 9
32 bilangan4 = 11
33 hasilPangkat = bilangan3 ** bilangan4
34 print(hasilPangkat)
35 #Pembagian Bulat
36 print(17//3)
37 #17 dibagi 3 adalah 5.6666. Karena dibulatkan maka akan menghasilkan
    nilai 6

```

1.10.2 Operator Perbandingan

Operator perbandingan (*comparison operators*) digunakan untuk membandingkan suatu nilai dari masing-masing operan.

Tabel 1.3 Operator Perbandingan

Operator	Contoh	Penjelasan
Tidak sama dengan <>	4 <> 4 bernilai False	Akan menghasilkan nilai kebalikan dari kondisi sebenarnya
Sama dengan ==	2 == 2 bernilai True	Akan menghasilkan nilai kebalikan dari kondisi sebenarnya
Tidak sama dengan !=	3 != 3 bernilai False	Akan menghasilkan nilai kebalikan dari kondisi sebenarnya
Lebih kecil dari <	7 < 5 bernilai True	Jika nilai operan kiri lebih kecil dari nilai operan kanan, maka kondisi menjadi benar
Lebih besar dari >	7 > 5 bernilai True	Jika nilai operan kiri lebih besar dari nilai operan kanan, maka kondisi menjadi benar
Lebih kecil atau sama dengan <=	7 > 5 bernilai True	Jika nilai operan kiri lebih kecil dari nilai operan kanan, atau sama, maka kondisi menjadi benar
Lebih besar atau sama dengan >=	5 >= 3 bernilai True	Jika nilai operan kiri lebih besar dari nilai operan kanan, atau sama, maka kondisi menjadi benar

1.10.3 Assignment Operator

Operator penugasan digunakan untuk memberikan atau memodifikasi nilai ke dalam sebuah variabel.

Tabel 1.4 Assignment Operator

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

1.10.4 Logical Operator

Tabel 1.5 Logical Operator

Operator	Contoh	Penjelasan
and	a, b = True, True # hasil akan True print a and b	Jika kedua operan bernilai True, maka kondisi akan bernilai True. Selain kondisi tadi maka akan bernilai False
or	a, b = True, False # hasil akan True print a or b print b or a print a or a # hasil akan False print b or b	Jika salah satu atau kedua operan bernilai True maka kondisi akan bernilai True. Jika keduanya False maka kondisi akan bernilai False
not	a, b = True, False # hasil akan True print not a print not b	Membalikkan nilai kebenaran pada operan misal jika asalnya True akan menjadi False dan begitupun sebaliknya

1.10.5 Bitwise Operator

Tabel 1.6 Bitwise Operator

Operator	Contoh	Penjelasan
&	a, b = 13, 37 # a akan bernilai '0000 1101' print a and b a, b = True, False # hasil akan True # b akan bernilai '0010 0101' c = a & b # c akan bernilai 5 = '0000 0101' print c	Operator biner AND, memeriksa akan bernilai True. Selain kondisi tadi maka akan bernilai False Jika salah satu atau kedua operan bernilai True apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika operasi akan bernilai 1
	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' c = a b # c akan bernilai 45 = '0010 1101' print c	Operator biner OR, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika salah satunya bernilai 1 maka bit hasil operasi akan bernilai 1
^	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' c = a ^ b # c akan bernilai 40 = '0010 1000'	Operator biner XOR, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika keduanya bernilai 1 maka bit hasil
Kali sama dengan *= a	a *= 2	Operator biner Negative, membalik nilai bit. Misal dari 1 menjadi 0, dari 0 menjadi 1
~	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101'	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan
<<	a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' # hasil bernilai 52 = '0011 0100' print a << 2 # hasil bernilai 148 = '1001 0100' print b << 2	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan. Yang diambil nantinya adalah sisa baginya

Tabel 1.7 Lanjutan Tabel Bitwise Operator

>>	<pre> a, b = 13, 37 # a akan bernilai '0000 1101' # b akan bernilai '0010 0101' # hasil bernilai 3 = '0000 0011' print a >> 2 # hasil bernilai 9 = '0000 1001' print b >> 2 </pre>	Operator penggeser biner ke kiri, deret bit akan digeser ke kiri sebanyak n kali
----	--	--

1.10.6 Membership Operator

Tabel 1.8 Membership Operator

in	<pre> sebuah_list = [1, 2, 3,4 ,5] print 5 in sebuah_list </pre>	Memeriksa apakah nilai yang dicari berada pada list atau struktur data python lainnya. Jika nilai tersebut ada maka kondisi akan bernilai True
not in	<pre> sebuah_list = [1, 2, 3,4 ,5] print 10 not in sebuah_list </pre>	Memeriksa apakah nilai yang dicari tidak ada pada list atau struktur data python lainnya. Jika nilai tersebut tidak ada maka kondisi akan bernilai True

1.10.7 Identity Operator

Tabel 1.9 Identity Operator

is	<pre> a, b = 10, 10 # hasil akan True print a is b </pre>	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang sama dengan nilai di sebelah kanan operan. Jika sama maka kondisi bernilai True.
is not	<pre> a, b = 10, 5 # hasil akan True print a is not b </pre>	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang berbeda dengan nilai di sebelah kanan operan. Jika berbeda maka kondisi bernilai True

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

```
1 #Kondisi if adalah kondisi yang akan dieksekusi oleh program jika bernilai
2 benar atau TRUE
3 nilai = 9
4 #jika kondisi benar/TRUE maka program akan mengeksekusi perintah
5 dibawahnya
6 if(nilai > 7):
7     print("Selamat Anda Lulus")
8 #jika kondisi salah/FALSE maka program tidak akan mengeksekusi
9 perintah
10 dibawahnya
11 if(nilai > 10):
12     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:

```
1 #Kondisi if else adalah jika kondisi bernilai TRUE maka akan
2 dieksekusi pada
3 if, tetapi jika bernilai FALSE maka akan dieksekusi kode pada else
4 nilai = 3
5 #Jika pernyataan pada if bernilai TRUE maka if akan dieksekusi,
6 tetapi jika
7 FALSE kode pada else yang akan dieksekusi.
8 if(nilai > 7):
9     print("Selamat Anda Lulus")
10 else :
11     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:

```
1 #Contoh penggunaan kondisi elif
2 hari_ini = "Minggu"
3 if(hari_ini == "Senin"):
4     print("Saya akan kuliah")
5 elif(hari_ini == "Selasa"):
6     print("Saya akan kuliah")
7 elif(hari_ini == "Rabu"):
8     print("Saya akan kuliah")
9 elif(hari_ini == "Kamis"):
10    print("Saya akan kuliah")
11 elif(hari_ini == "Jumat"):
12    print("Saya akan kuliah")
13 elif(hari_ini == "Sabtu"):
14    print("Saya akan kuliah")
15 elif(hari_ini == "Minggu"):
16    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

1.14.1 Pengulangan While

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

```
1 #Contoh penggunaan While Loop
2 count = 0
3 while (count < 9):
4     print ('The count is:', count)
5     count = count + 1
6 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:

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

1.16 Pengulangan Bersarang (Nested Loop)

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

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

1.17 Number Python

Number adalah tipe data Python yang menyimpan nilai numerik. Number adalah tipe data yang tidak berubah. Ini berarti, mengubah nilai dari sejumlah tipe data

akan menghasilkan objek yang baru dialokasikan. Objek Number dibuat saat Anda memberikan nilai pada-nya. Sebagai contoh : `angkaPertama = 1` `angkaKedua = 33`

Python mendukung beberapa tipe data Number diantaranya :

- Int
- Float
- Complex

Berikut ini adalah beberapa contoh dari Tipe data Number pada Python :

Tabel 1.10 Tipe Data Number Pada Python

Int	Float	Complex
20	0.1	3.14j
300	1.20	35.j
-13	-41.2	3.12e-12j
020	32.23+e123	.873j
-0103	-92.	-.123+0J
-0x212	-32.52e10	3e+123J
0x56	60.2-E13	4.31e-4j

1.17.1 Konversi Tipe Data Number Python

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

- `int(x)`
untuk meng-konversi x menjadi plain integer.
- `ong(x)`
untuk meng-konversi x menjadi long integer.
- `float(x)`
untuk meng-konversi x menjadi floating point number.
- `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.

1.17.2 Fungsi Matematika

Pada bahasa pemrograman Python terdapat fungsi untuk melakukan perhitungan matematis, berikut adalah daftarnya :

Tabel 1.11 Fungsi Matematika

Nama	Pengguna	Penjelasan
Absolute	<code>abs(x)</code>	Nilai absolut dari x: (positive) jarak antara x and 0.
Ceiling	<code>ceil(x)</code>	Ceiling dari x: integer terkecil yang kurang dari x.
Cmp	<code>cmp(x, y)</code>	-1 if $x < y$, 0 if $x == y$, or 1 if $x > y$. Tidak berlaku lagi dengan Python 3. Sebaliknya gunakan <code>return (x>y)-(x</code>
Eksponen	<code>exp(x)</code>	Nilai eksponen dari x: e^x
Fabs	<code>fabs(x)</code>	Nilai absolut dari x.
Floor	<code>floor(x)</code>	Nilai dasar dari x: integer terbesar tidak lebih besar dari x
Log	<code>log(x)</code>	Logaritma dari x, untuk $x > 0$.
Log 10	<code>log10(x)</code>	Basis 10 logaritma dari x, untuk $x > 0$
Max	<code>max(x1, x2,...)</code>	Argumen terbesar: Nilai terdekat dengan tak terhingga positif
Min	<code>min(x1, x2,...)</code>	Argumen terkecil: nilai yang paling mendekati tak berhingga negatif.
Modf	<code>modf(x)</code>	Bagian pecahan dan bilangan bulat dari x dalam tuple dua item. Kedua bagian memiliki tanda yang sama dengan x. Bagian integer dikembalikan sebagai float.
Pow	<code>pow(x, y)</code>	Nilai $x ** y$.
Round	<code>round(x [,n])</code>	X dibulatkan menjadi n digit dari titik desimal. Putaran Python jauh dari nol sebagai tie-breaker: <code>round (0.5)</code> adalah 1.0 dan <code>round (-0.5)</code> adalah -1.0.
Akar Kuadrat	<code>sqrt(x)</code>	Akar kuadrat x untuk $x > 0$.

1.18 Fungsi Nomor Acak

Nomor acak digunakan untuk aplikasi permainan, simulasi, pengujian, keamanan, dan privasi. Python mencakup fungsi berikut yang umum digunakan. Berikut adalah daftarnya :

Tabel 1.12 Fungsi Nomor Acak

Nama	Penggunaan	Penjelasan
Choice	choice(seq)	Item acak dari list, tuple, atau string.
RandRange	randrange ([start,] stop [,step])	Elemen yang dipilih secara acak dari jangkauan (start, stop, step).
Random	random()	A random float r, sehingga 0 kurang dari atau sama dengan r dan r kurang dari 1
Seed	seed([x])	Menetapkan nilai awal integer yang digunakan dalam menghasilkan bilangan acak. Panggil fungsi ini sebelum memanggil fungsi modul acak lainnya. Tidak ada pengembalian
Shuffle	shuffle(lst)	Mengacak daftar dari daftar di tempat. Tidak ada pengembalian
Floor	floor(x)	The floor of x: the largest integer not greater than x.
Uniform	uniform(x, y)	Sebuah float acak r, sedemikian rupa sehingga x kurang dari atau sama dengan r dan r kurang dari y.

1.18.1 Fungsi Trigonometri

Python mencakup fungsi berikut yang melakukan perhitungan trigonometri. Berikut adalah daftarnya :

Tabel 1.13 Fungsi Trigonometri

Nama	Penggunaan	Penjelasan
Acos	acos(x)	Kembalikan kosinus x, di radian.
Asin	asin(x)	Kembalikan busur sinus x, dalam radian.
Atan	atan(x)	Kembalikan busur singgung x, di radian.
Atan 2	atan2(y, x)	Kembali atan (y / x), di radian
Kosinus	cos(x)	Kembalikan kosinus x radian.
Hypot	hypot(x, y)	Kembalikan norma Euclidean, $\sqrt{x^2 + y^2}$.
Sin	sin(x)	Kembalikan sinus dari x radian.
Tan	tan(x)	Kembalikan tangen x radian.
Derajat	degrees(x)	Mengonversi sudut x dari radian ke derajat.
Radian	radians(x)	Mengonversi sudut x dari derajat ke radian.

1.18.2 Konstanta Matematika

Modul ini juga mendefinisikan dua konstanta matematika. Berikut adalah daftarnya :

Tabel 1.14 Konstanta Matematika

Nama	Penggunaan	Penjelasan
Pi	pi	Konstanta Pi matematika
e	e	Konstanta e matematika

1.19 String

String adalah jenis yang paling populer di bahasa pemrograman. Kita bisa membuatnya hanya dengan melampirkan karakter dalam tanda kutip. Python memperlakukan tanda kutip tunggal sama dengan tanda kutip ganda. Membuat string semudah memberi nilai pada sebuah variabel.

Dibawah ini adalah contoh sederhana dari sebuah string pada bahasa pemrograman Python.

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

Python tidak menggunakan tipe karakter titik koma ; Ini diperlakukan sebagai string dengan panjang satu, sehingga juga dianggap sebagai substring.

Untuk mengakses substring, gunakan tanda kurung siku untuk mengiris beserta indeks atau indeks untuk mendapatkan substring Anda. Sebagai contoh :

```
1 name = 'Esi Vidia' message = "Esi harus tetap semangat dalam belajar"
2 print ("name[0]: ", name[0])
3 print ("message[1:4]: ", message[1:3])
```

Bila kode diatas dieksekusi, maka akan menghasilkan hasil sebagai berikut :

name[0]: E

message[1:4]: si

1.19.1 Mengupdate String

Anda dapat "memperbarui" string yang ada dengan (kembali) menugaskan variabel ke string lain. Nilai baru dapat dikaitkan dengan nilai sebelumnya atau ke string yang sama sekali berbeda sama sekali. Sebagai contoh

```
1 message = 'Hello World'
2 print ("Updated String :- ", message[:6] + 'Python')
```

Bila kode diatas dieksekusi, maka akan menghasilkan hasil sebagai berikut :

Updated String:- Hello Python

1.19.2 Escape Character

Dibawah ini adalah tabel dari daftar karakter escape atau karakter non-printable yang dapat diwakili/ditulis dengan awalan notasi backslash.

Tabel 1.15 Escape Character

Notasi Backslash	Karakter Hexadecimal	Penjelasan
a	0x07	Bell atau alert
b	0x08	Backspace
cx		Control-x
C-x		Control-x
e	0x1b	Escape
f	0x0c	Formfeed
M- C-x		Meta-Control-x
n	0x0a	Newline
nnn		Octal notation, dimana n berada di range 0.7
r	0x0d	Carriage return
s	0x20	Space
t	0x09	Tab
v	0x0b	Vertical tab
x		Character x
xnn	asdafsdfsdf	Notasi Hexadecimal, dimana n berada di range 0.9, a.f, atau A.F

1.19.3 Operator Special String

Asumsikan variabel string adalah 'Belajar' dan variabel b adalah 'Python', lalu dibawah ini adalah operator yang bisa dipakai pada kedua string di variabel tersebut.

a = "Belajar"

b = "Python"

Berikut adalah daftar operator spesial string pada Python :

Tabel 1.16 daftar operator spesial string

Operator	Contoh	Penjelasan
+	a + b akan menghasilkan BelajarPython	Concatenation - Menambahkan nilai pada kedua sisi operator
*	a*2 akan menghasilkan BelajarBelajar	Pengulangan - Membuat string baru, menggabungkan beberapa salinan dari string yang sama
[]	a[1] akan menghasilkan e	Slice - Memberikan karakter dari indeks yang diberikan
[:]	a[1:4] akan menghasilkan ela	Range Slice - Memberikan karakter dari kisaran yang diberikan
in	B in a akan menghasilkan 1	Keanggotaan - Mengembalikan nilai true jika ada karakter dalam string yang diberikan
not in	Z not in a akan menghasilkan 1	Keanggotaan - Mengembalikan nilai true jika karakter tidak ada dalam string yang diberikan
r/R	print r'n' prints \n dan print R'	Raw String - Menekan arti aktual karakter Escape. Sintaks untuk string mentah sama persis dengan string biasa kecuali operator string mentah, huruf "r", yang mendahului tanda petik. "R" bisa berupa huruf kecil (r) atau huruf besar (R) dan harus ditempatkan tepat sebelum tanda kutip pertama.
%		Format - Melakukan format String

1.19.4 Operator Format String

Salah satu fitur Python yang paling keren adalah format string operator `%`. Operator ini unik untuk string dan membuat paket memiliki fungsi dari keluarga `printf C ()`. Berikut adalah contoh sederhananya :

```
print ("My name is %s and weight is %d kg!" % ('Zara', 21))
```

Berikut adalah daftar lengkap simbol yang bisa digunakan bersamaan dengan `%` :

Tabel 1.17 daftar simbol

%c	character
%s	Konversi string melalui <code>str ()</code> sebelum memformat
%i	Dianggap sebagai bilangan bulat desimal
%d	Dianggap sebagai bilangan bulat desimal
%u	Unsigned decimal integer
%o	Bilangan bulat oktal
%x	Bilangan bulat heksadesimal (huruf kecil)
%X	Bilangan bulat heksadesimal (huruf besar)
%e	Notasi eksponensial (dengan huruf kecil 'e')
%E	Notasi eksponensial (dengan huruf besar 'E')
%f	Bilangan real floating point
%g	Yang lebih pendek dari <code>% f</code> dan <code>% e</code>
%G	Lebih pendek dari <code>% f</code> dan <code>% E</code>

1.19.5 Triple Quote

Python triple quotes digunakan dengan membiarkan string untuk ditulis dalam beberapa baris, termasuk kata kerja `NEWLINEs`, `TABs`, dan karakter khusus lainnya. Sintaks untuk triple quotes terdiri dari tiga tanda kutip tunggal atau ganda ditulis berturut-turut.

Berikut adalah contohnya :

```
1 kutipantiga = """this is a long string that is made up of
2 several lines and non-printable characters such as
3 TAB ( \t ) and they will show up that way when displayed.
4 NEWLINEs within the string, whether explicitly given like
5 this within the brackets [ \n ], or just a NEWLINE within
6 the variable assignment will also show up.
7 """
8 print (kutipantiga)
```

1.19.6 String Unicode

Pada Python 3, semua string diwakili dalam Unicode. Sedangkan pada Python 2 disimpan secara internal sebagai 8-bit ASCII, maka diperlukan lampiran 'u' untuk membuatnya menjadi Unicode. Tetapi hal ini tidak lagi diperlukan sekarang.

Metode String Built-in

Python menyertakan metode built-in berikut untuk memanipulasi string

Tabel 1.18 Metode built-in

Metode	Penjelasan
capitalize()	Meng-kapitalkan huruf pertama string
center(width, fillchar)	Mengembalikan string yang dilapisi dengan fillchar dengan string asli yang dipusatkan pada total width kolom.
count(str, beg = 0, end = len(string))	Menghitung berapa kali str yang terjadi dalam string atau dalam substring string jika memulai indeks beg dan end index end diberikan.
decode(encoding = 'UTF 8', errors = 'strict')	Dekode string menggunakan codec yang terdaftar untuk pengkodean. Encoding default ke pengkodean string default.
encode(encoding = 'UTF 8', errors = 'strict')	Mengembalikan versi string yang dikodekan string; Pada kesalahan, default adalah menaikkan ValueError kecuali jika kesalahan diberikan dengan 'ignore' atau 'replace'.
endswith(suffix, beg = 0, end = len(string))	Menentukan apakah string atau substring string (jika memulai indeks memohon dan mengakhiri akhir indeks diberikan) berakhir dengan akhiran; Mengembalikan nilai true jika benar dan salah.
expandtabs(tabsize = 8)	Memperluas tab dalam string ke banyak ruang; Default ke 8 spasi per tab jika tabsize tidak tersedia.
find(str, beg = 0 end = len(string))	Tentukan jika str terjadi dalam string atau dalam substring string jika memulai indeks beg dan end index end diberikan return index jika ditemukan dan -1 sebaliknya
index(str, beg = 0, end = len(string))	Sama seperti find (), namun menimbulkan pengecualian jika str tidak ditemukan.
isalnum()	Mengembalikan true jika string memiliki minimal 1 karakter dan semua karakternya alfanumerik dan false sebaliknya.

Tabel 1.19 Lanjutan Tabel Metode built-in

isalpha()	Mengembalikan true jika string memiliki minimal 1 karakter dan semua karakter adalah abjad dan false sebaliknya.
isdigit()	Mengembalikan true jika string hanya berisi digit dan false sebaliknya.
islower()	Mengembalikan true jika string memiliki setidaknya 1 karakter casing dan semua karakter casing dalam huruf kecil dan false sebaliknya
isnumeric()	Mengembalikan true jika string unicode hanya berisi karakter numerik dan false sebaliknya.
isspace()	Mengembalikan true jika string hanya berisi karakter spasi dan false sebaliknya.
istitle()	Mengembalikan true jika string benar "titlecased" dan false sebaliknya.
isupper()	Mengembalikan true jika string memiliki setidaknya satu karakter casing dan semua karakter casing ada dalam huruf besar dan false sebaliknya.
join(seq)	Merges (concatenates) representasi string elemen dalam urutan seq menjadi string, dengan string pemisah.
len(string)	Mengembalikan panjang string
ljust(width[, fillchar])	Mengembalikan string berlapis ruang dengan string asli dibiarkan dibenarkan ke kolom lebar total
lower()	Mengonversi semua huruf besar dalam bentuk string menjadi huruf kecil.
lstrip()	Menghapus semua spasi utama dalam string.
maketrans()	Mengembalikan tabel terjemahan untuk digunakan dalam fungsi terjemahan.
max(str)	Mengembalikan karakter alfabetik dari string str
min(str)	Mengembalikan min karakter abjad dari string str.
replace(old, new [, max])	Menggantikan semua kemunculan lama dalam string dengan kejadian baru atau paling maksimal jika max diberikan.
rfind(str, beg = 0, end = len(string))	Sama seperti find (), tapi cari mundur dalam string.
rindex(str, beg = 0, end = len(string))	Sama seperti index (), tapi cari mundur dalam string.

Tabel 1.20 Lanjutan Tabel Metode built-in

rjust(width[, fillchar])	Mengembalikan string berlapis ruang dengan senar asli benar-dibenarkan untuk total kolom lebar.
rstrip()	Menghapus semua spasi spasi string.
split(str="", num=string.count(str))	Membagi string sesuai dengan pemisah str (ruang jika tidak disediakan) dan mengembalikan daftar
	substring; Terpecah menjadi paling banyak substring
	jika diberikan.
splitlines(num=string.count("\n"))	Membagi string sama sekali (atau num) NEWLINES dan mengembalikan daftar setiap baris dengan NEWLINES dihapus
startswith(str, beg=0,end=len(string))	Determines if string or a substring of string (if starting index beg and ending index end are given) starts with substring str; returns true if so and false otherwise.
strip([chars])	Lakukan kedua lstrip () dan rstrip () pada string
swapcase()	Kasus invers untuk semua huruf dalam string.
title()	Mengembalikan versi string "titlecased", yaitu, semua kata diawali dengan huruf besar dan sisanya huruf kecil.
translate(table, deletechars="")	Menerjemahkan string sesuai dengan tabel terjemahan str (256 karakter), menghapus string del.
upper()	Mengonversi huruf kecil dalam bentuk string ke huruf besar.
zfill (width)	Mengembalikan string asli yang tertinggal dengan angka nol ke total karakter lebar; Dimaksudkan untuk angka, zfill () mempertahankan tanda apapun yang diberikan (kurang satu nol).
isdecimal()	Mengembalikan nilai true jika string unicode hanya berisi karakter desimal dan false sebaliknya.

1.20 List

Dalam bahasa pemrograman Python, struktur data yang paling dasar adalah urutan atau lists. Setiap elemen-elemen berurutan akan diberi nomor posisi atau indeks. Indeks pertama dalam list adalah nol, indeks kedua adalah satu dan seterusnya.

Python memiliki enam jenis urutan built-in, namun yang paling umum adalah list dan tuple. Ada beberapa hal yang dapat Anda lakukan dengan semua jenis list. Operasi ini meliputi pengindeksan, pengiris, penambahan, perbanyak, dan pengecekan keanggotaan. Selain itu, Python memiliki fungsi built-in untuk menemukan panjang list dan untuk menemukan elemen terbesar dan terkecilnya.

1.20.1 Membuat List Python

List adalah tipe data yang paling serbaguna yang tersedia dalam bahasa Python, yang dapat ditulis sebagai daftar nilai yang dipisahkan koma (item) antara tanda kurung siku. Hal penting tentang daftar adalah item dalam list tidak boleh sama jenisnya.

Membuat list sangat sederhana, tinggal memasukkan berbagai nilai yang dipisahkan koma di antara tanda kurung siku. Dibawah ini adalah contoh sederhana pembuatan list dalam bahasa Python.

```
1 #Contoh sederhana pembuatan list pada bahasa pemrograman python
2 list1 = [ 'kimia', 'fisika', 1993, 2017 ]
3 list2 = [ 1, 2, 3, 4, 5 ]
4 list3 = [ "a", "b", "c", "d" ]
```

1.20.2 Akses Nilai Dalam List

Untuk mengakses nilai dalam list python, gunakan tanda kurung siku untuk mengiris beserta indeks atau indeks untuk mendapatkan nilai yang tersedia pada indeks tersebut.

Berikut adalah contoh cara mengakses nilai di dalam list python :

```
1 #Cara mengakses nilai di dalam list Python
2 list1 = [ 'fisika', 'kimia', 1993, 2017 ]
3 list2 = [ 1, 2, 3, 4, 5, 6, 7 ]
4 print ("list1[0]: ", list1[0])
5 print ("list2[1:5]: ", list2[1:5])
```

Setelah Anda mengeksekusi kode diatas, hasilnya akan seperti dibawah ini :

list1[0]: fisika

list2[1:5]: [2, 3, 4, 5]

1.20.3 Update Nilai Dalam List

Anda dapat memperbarui satu atau beberapa nilai di dalam list dengan memberikan potongan di sisi kiri operator penugasan, dan Anda dapat menambahkan nilai ke dalam list dengan metode `append()`. Sebagai contoh :

```
1 list = ['fisika', 'kimia', 1993, 2017]
2 print ("Nilai ada pada index 2 : ", list[2])
3 list[2] = 2001
4 print ("Nilai baru ada pada index 2 : ", list[2])
```

1.20.4 Hapus Nilai Dalam List

Untuk menghapus nilai di dalam list python, Anda dapat menggunakan salah satu pernyataan **del** jika Anda tahu persis elemen yang Anda hapus. Anda dapat menggunakan metode **remove()** jika Anda tidak tahu persis item mana yang akan dihapus. Sebagai contoh :

```
1 #Contoh cara menghapus nilai pada list python
2 list = ['fisika', 'kimia', 1993, 2017]
3 print (list)
4 del list[2]
5 print ("Setelah dihapus nilai pada index 2 : ", list)
```

1.21 Operasi Dasar

List Python merespons operator `+` dan `*` seperti string; Itu artinya penggabungan dan pengulangan di sini juga berlaku, kecuali hasilnya adalah list baru, bukan sebuah String. Sebenarnya, list merespons semua operasi urutan umum yang kami gunakan pada String di bab sebelumnya. Dibawah ini adalah tabel daftar operasi dasar pada list python.

Tabel 1.21 Operasi Dasar

Python Expression	Hasil	Penjelasan
<code>len([1, 2, 3, 4])</code>	4	Length
<code>[1, 2, 3] + [4, 5, 6]</code>	<code>[1, 2, 3, 4, 5, 6]</code>	Concatenation
<code>['Halo!'] * 4</code>	<code>['Halo!', 'Halo!', 'Halo!', 'Halo!']</code>	Repetition
<code>2 in [1, 2, 3]</code>	True	Membership
<code>for x in [1,2,3] :</code>	1 2 3	Iteration
<code>print (x,end = ' ')</code>		

1.21.1 Indexing, Slicing dan Matrix pada List Python

Karena list adalah urutan, pengindeksan dan pengiris bekerja dengan cara yang sama untuk list seperti yang mereka lakukan untuk String.

Dengan asumsi input berikut :

L = ['C++', 'Java', 'Python']

Tabel 1.22 Indexing, Slicing dan Matrix pada List Python

Python Expression	Hasil	Penjelasan
L[2]	'Python'	Offset mulai dari nol
L[-2]	'Java'	Negatif: hitung dari kanan
L[1:]	['Java', 'Python']	Slicing mengambil bagian

1.21.2 Method dan Fungsi Build-in pada List Python

Python menyertakan fungsi built-in sebagai berikut

Tabel 1.23 Fungsi built-in

Python Function	Penjelasan
cmp(list1, list2)	# Tidak lagi tersedia dengan Python 3
len(list)	Memberikan total panjang list.
max(list)	Mengembalikan item dari list dengan nilai maks
min(list)	Mengembalikan item dari list dengan nilai min.
list(seq)	Mengubah tuple menjadi list.

Python menyertakan methods built-in sebagai berikut:

Tabel 1.24 Methods built-in

Python Method	Penjelasan
list.append(obj)	Menambahkan objek obj ke list
list.count(obj)	Jumlah pengembalian berapa kali obj terjadi dalam list
list.extend(seq)	Tambahkan isi seq ke list
list.index(obj)	Mengembalikan indeks terendah dalam list yang muncul obj
list.insert(index, obj)	Sisipkan objek obj ke dalam list di indeks offset
list.pop(obj = list[-1])	Menghapus dan mengembalikan objek atau obj terakhir dari list
list.remove(obj)	Removes object obj from list
list.reverse()	Membalik list objek di tempat
list.sort([func])	Urutkan objek list, gunakan compare func jika diberikan

1.22 Tuple

Sebuah tuple adalah urutan objek Python yang tidak berubah. Tuple adalah urutan, seperti daftar. Perbedaan utama antara tuple dan daftarnya adalah bahwa tuple tidak dapat diubah tidak seperti List Python. Tuple menggunakan tanda kurung, sedangkan List Python menggunakan tanda kurung siku.

Membuat tuple semudah memasukkan nilai-nilai yang dipisahkan koma. Secara opsional, Anda dapat memasukkan nilai-nilai yang dipisahkan koma ini di antara tanda kurung juga. Sebagai contoh :

```

1 #Contoh sederhana pembuatan tuple pada bahasa pemrograman python
2 tup1 = ( 'fisika' , 'kimia' , 1993 , 2017 )
3 tup2 = (1 , 2 , 3 , 4 , 5 )
4 tup3 = "a" , "b" , "c" , "d"

```

Tuple kosong ditulis sebagai dua tanda kurung yang tidak berisi apa-apa, contohnya :

```
tup1 = ();
```

Untuk menulis tuple yang berisi satu nilai, Anda harus memasukkan koma, meskipun hanya ada satu nilai, contohnya :**tup1 = (50,)**

Seperti indeks String, indeks tuple mulai dari 0, dan mereka dapat diiris, digabungkan, dan seterusnya.

1.22.1 Akses Nilai Dalam Tuple

Untuk mengakses nilai dalam tuple, gunakan tanda kurung siku untuk mengiris beserta indeks atau indeks untuk mendapatkan nilai yang tersedia pada indeks tersebut. Sebagai contoh :

```
1 #Cara mengakses nilai tuple
2 tup1 = ('fisika', 'kimia', 1993, 2017)
3 tup2 = (1, 2, 3, 4, 5, 6, 7)
4 print ("tup1[0]: ", tup1[0])
5 print ("tup2[1:5]: ", tup2[1:5])
```

Setelah Anda mengeksekusi kode diatas, hasilnya akan seperti dibawah ini :

tup1[0]: fisika

tup2[1:5]: (2, 3, 4, 5)

1.23 Update Nilai Dalam Tuple

Tuple tidak berubah, yang berarti Anda tidak dapat memperbarui atau mengubah nilai elemen tuple. Anda dapat mengambil bagian dari tuple yang ada untuk membuat tuple baru seperti ditunjukkan oleh contoh berikut.

```
1 tup1 = (12, 34.56)
2 tup2 = ('abc', 'xyz')
3 # Aksi seperti dibawah ini tidak bisa dilakukan pada tuple python
4 # Karena memang nilai pada tuple python tidak bisa diubah
5 # tup1[0] = 100;
6 # Jadi, buatlah tuple baru sebagai berikut
7 tup3 = tup1 + tup2
8 print (tup3)
```

1.24 Menghapus Nilai Dalam Tuple

Menghapus elemen tuple individual tidak mungkin dilakukan. Tentu saja, tidak ada yang salah dengan menggabungkan tuple lain dengan unsur-unsur yang tidak diinginkan dibuang.

Untuk secara eksplisit menghapus keseluruhan tuple, cukup gunakan del statement. Sebagai contoh

```
1 tup = ('fisika', 'kimia', 1993, 2017);
2 print (tup)
3 del tup;
4 print "Setelah menghapus tuple : "
5 print tup
```

1.25 Operasi Dasar Pada List Tuple

Tupel merespons operator + dan * sama seperti String; Mereka berarti penggabungan dan pengulangan di sini juga berlaku, kecuali hasilnya adalah tupel baru, bukan string.

Sebenarnya, list merespons semua operasi urutan umum yang kami gunakan pada String di bab sebelumnya. Dibawah ini adalah tabel daftar operasi dasar pada list python.

Tabel 1.25 daftar operasi dasar pada python

Python Expression	Hasil	Penjelasan
<code>len((1, 2, 3))</code>	3	Length
<code>(1, 2, 3) + (4, 5, 6)</code>	<code>(1, 2, 3, 4, 5, 6)</code>	Concatenation
<code>('Halo!') * 4</code>	<code>('Halo!', 'Halo!', 'Halo!', 'Halo!')</code>	Repetition
<code>3 in (1, 2, 3)</code>	True	Membership
<code>for x in (1,2,3) : print (x, end = ' ')</code>	1 2 3	Iteration

1.26 Indexing, Slicing dan Matrix

Karena tupel adalah urutan, pengindeksan dan pengiris bekerja dengan cara yang sama untuk tupel seperti pada String, dengan asumsi masukan berikut

Dengan asumsi input berikut :

T = ('C++', 'Java', 'Python')

Tabel 1.26 Indexing, Slicing dan Matrix

Python Expression	Hasil	Penjelasan
<code>T[2]</code>	<code>'Python'</code>	Offset mulai dari nol
<code>T[-2]</code>	<code>'Java'</code>	Negatif: hitung dari kanan
<code>T[1:]</code>	<code>('Java', 'Python')</code>	Slicing mengambil bagian

1.27 Fungsi Build-in

Python menyertakan fungsi built-in sebagai berikut

Tabel 1.27 fungsi built-in

Python Function	Penjelasan
cmp(tuple1, tuple2)	# Tidak lagi tersedia dengan Python 3
len(tuple)	Memberikan total panjang tuple.
max(tuple)	Mengembalikan item dari tuple dengan nilai maks.
min(tuple)	Mengembalikan item dari tuple dengan nilai min.
tuple(seq)	Mengubah tuple menjadi tuple.

1.28 Dictionary Python

Dictionary Python berbeda dengan List ataupun Tuple. Karena setiap urutannya 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 tuple.

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

```
1 #Contoh cara membuat Dictionary pada Python
2 dict = {'Name': 'Dava', 'Age': 12, 'Class': 'SMP'}
3 print ("dict['Name']: ", dict['Name'])
4 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.

```
1 #Update dictionary python
2 dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
3 dict['Age'] = 8; # Mengubah entri yang sudah ada
4 dict['School'] = "DPS School" # Menambah entri baru
5 print ("dict['Age']: ", dict['Age'])
6 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 :

```

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

```

1.29 Tanggal dan Jam

Program Python yang dapat menangani tanggal dan waktu dalam beberapa cara. Mengkonversi antara format tanggal adalah tugas umum untuk komputer. Modul Python's waktu dan kalender membantu melacak tanggal dan waktu.

Interval waktu adalah angka floating-point dalam satuan detik. Instants tertentu dalam waktu dinyatakan dalam satu detik sejak 12:00 am, 1 Januari 1970(epoch).

Ini adalah waktu yang populer modul yang tersedia di Python yang menyediakan fungsi untuk bekerja dengan waktu, dan untuk mengkonversi antara pernyataan. Fungsi `time.time()` mengembalikan sistem saat ini waktu sejak 12:00 am, 1 Januari 1970.

```

1 import time; # harus menginclude modul time
2 ticks = time.time()
3 print "Number of ticks since 12:00am, January 1, 1970:", ticks

```

Bila kode diatas dieksekusi, maka akan menghasilkan hasil sebagai berikut :

Number of ticks since 12:00am, January 1, 1970: 7186862.73399

1.30 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:

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


BAB 2

BIG DATA

2.1 Sejarah Big Data

Sejarah Big Data yaitu yang bermula dari sejarahnya Apache Hadoop. Doug Cutting adalah orang yang iseng-iseng membuat search engine. Doug Cutting menemukan beberapa permasalahan karena search engine milik crawler (untuk ngumpulin web page di internet), indexer, storage dan search algo storage nya harus besar dan tidak bisa hanya disimpan di dalam satu node saja karena disetiap saat terdapat kemungkinan network failure, power failure, node failure dan segala macam failure lain nya yang ada di distributed system. Doug Cutting stuck, bagaimana cara nya agar menyimpan data sebesar itu dan juga cara processing nya, dan bagaimana agar efisien dari data yang tersebar itu sampai akhirnya Doug Cutting menemukan solusi dari permasalahan itu karena google mengeluarkan 2 buah paper Google File System dan juga MapReduce. Fenomena Big Data, dimulai pada tahun 2000-an ketika seorang analis industri Doug menyampaikan konsep Big Data yang terdiri dari tiga bagian penting yaitu 3V (Volume, Velocity, Variety).

2.2 Gambaran Big Data

Dipandang dari sudut pandang ilmu pengetahuan, media penyimpanan yang terdapat pada hardware yang dapat digunakan adalah HDD, FDD, dan juga yang sejenisnya. Sedangkan media penyimpanan pada jaringan biologi, pada diri kita dikaruniai otak oleh Sang Pencipta. Seberapa penting mengolah data-data yang kecil kemudian berkumpul menjadi data yang besar atau bisa disebut dengan Big Data tersebut.



Gambar 2.1 Karakteristik Big Data 10V

Terdapat beberapa elemen penting dalam big data diantaranya:

1. Data (Facts, a description of the World)
2. Information (Captured Data and Knowledge): Merekam atau mengambil Data dan Knowledge pada satu waktu tertentu (at a single point). Sedangkan Data dan Knowledge dapat terus berubah dan bertambah dari waktu ke waktu.
3. Knowledge (Our personal map/model of the world): apa yang kita ketahui (not the real world itself) Anda saat ini tidak dapat menyimpan pengetahuan dalam diri anda dalam apa pun selain otak, dan untuk membangun pengetahuan perlu informasi dan data.

Menurut McKinsey Global (2011), Big Data bisa didefinisikan dengan data yang memiliki skala (volume), distribusi (velocity), keragaman (variety) yang sangatlah besar, dan atau abadi, sehingga sangat membutuhkan penggunaan arsitektur teknikal dan metode analitik yang inovatif untuk mendapatkan wawasan yang dapat memberikan nilai bisnis baru (informasi yang bermakna). Dan pada pengembangannya

ada yang menyebut (7V) termasuk Volume, Velocity, Variety, Variability, Veracity, Value, dan Visualization, atau 10V bahkan lebih dari itu.



Gambar 2.2 Big Data

Big data adalah sebuah istilah dari sekumpulan data yang begitu besar atau kompleks dimana tidak bisa ditangani lagi dengan menggunakan sistem teknologi komputer konvensional.

2.3 Karakteristik Big Data

1. Volume

Volume data yang akan terus meningkat dari waktu ke waktu. Serta banyak faktor yang mendukung meningkatnya volume data secara sangat pesat, diantaranya yaitu hampir semua transaksi bisnis melibatkan data, meningkatnya jumlah unstructured data yang mengalir dari media sosial, dan meningkatnya jumlah data yang dihasilkan dari mesin serta perangkat dari mobile.

- Facebook menghasilkan 10TB data baru setiap hari, Twitter 7TB
- Sebuah Boeing 737 menghasilkan 240 terabyte data penerbangan selama penerbangan dari satu wilayah bagian AS ke wilayah yang lain
- Microsoft kini memiliki satu juta server, kurang dari Google, tetapi lebih dari Amazon, kata Ballmer (2013).

Big Data mengarah pada manajemen informasi skala yang besar (large-scale) dan teknologi analisis yang melebihi kapabilitas teknologi data secara tradi-

sional. Ada perbedaan yang paling utama dalam tiga hal antara Tradisional dengan Big Data, yaitu the rate of data generation, amount of data (volume) and transmission (velocity), dan the types of structured and unstructured data (variety).



Gambar 2.3 Volume

Teknologi pada Big Data dibagi menjadi 2 kelompok, yaitu batch processing yang mana digunakan untuk menganalisis data yang sudah settle (data at rest) pada satu waktu tertentu. Dan streaming processing yang mana digunakan untuk menganalisis data yang terus menerus terupdate setiap waktu (data in motion).

2. Velocity

Velocity merupakan kecepatan data yang masuk baik dalam per jam, per detik, dan lainnya. Clickstreams (web log) dan transfer data asynchronous yang dapat menangkap apa saja yang dilakukan oleh jutaan atau lebih pengguna yang lakukan saat ini. Dimana clickstream atau web log merupakan salah satu sumber data yang menarik. Sebuah clickstream meliputi suatu rekaman untuk setiap permintaan halaman dari setiap pengunjung dari website. Oleh karena itu, suatu clickstream merekam setiap gesture yang dibuat oleh pengunjung dan gesture memiliki potensi untuk memberikan deskripsi mengenai kebiasaan dari pengunjung yang bersangkutan. Diharapkan bahwa clickstream akan mengidentifikasi sesi yang berhasil dan tidak berhasil, menentukan apakah pengunjung puas atau tidak puas, dan menemukan bagian dari website yang secara efektif menarik perhatian pengunjung.



Gambar 2.4 Velocity

3. Variety

Variety adalah sekumpulan dari berbagai macam-macam data, baik data yang terstruktur, semi terstruktur maupun data tidak terstruktur (bisa dipastikan lebih mendominasi). Tampilan data semakin komprehensif (lengkap dan menyeluruh).



Gambar 2.5 Variety

Saat ini data memiliki banyak format dan tipe. Data tersebut ada yang berupa structured data dan unstructured data. Informasi dihasilkan dari aplikasi bisnis yang digunakan oleh organisasi dan melibatkan structured data ataupun unstructured data. Mengelola, menggabungkan dan mengatur data yang memiliki beragam format dan tipe merupakan suatu tantangan yang harus diselesaikan

dan dijawab oleh suatu organisasi. Selain itu untuk menghasilkan pengetahuan dari data yang berjumlah besar tersebut perlu untuk menghubungkan semua data dari beragam tipe dan format.

2.4 Ekosistem Big Data Analytics

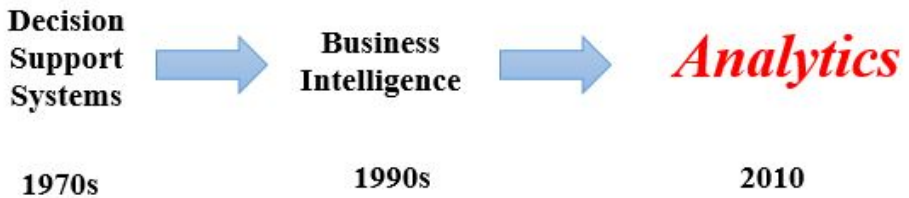


Gambar 2.6 Ekosistem Big Data

Keterangan:

1. Data Devices
2. Data Collectors
3. Data Aggregators: penggabungan beberapa informasi dari database dengan memiliki tujuan untuk mempersiapkan dataset gabungan untuk pengolahan data.
4. Data Users/ Buyers

Sebuah titik awal untuk memahami Analytics adalah cara untuk mengeksplorasi atau menyelidiki atau juga memahami secara mendalam suatu objek sampai ke akar-akarnya. Hasil analytics biasanya tidak menyebabkan banyak kebingungan, karena konteksnya biasanya membuat makna yang jelas. Perkembangan analytics dimulai dari DSS kemudian berkembang menjadi Business Intelligence baru kemudian menjadi analytics yang ditunjukkan oleh Gambar 2.7 berikut:



Gambar 2.7 Perkembangan Analytics

Bussines Intelligence bisa dilihat sebagai suatu istilah umum untuk semua aplikasi yang mendukung DSS, dan bagaimana hal itu dijelaskan dalam industri dan semakin meluas sampai di kalangan akademisi. Bussines Intelligence berevolusi dari DSS, dan orang dapat berargumentasi bahwa Analytics berevolusi dari Bussines Intelligence. Dengan demikian, Analytics merupakan istilah umum untuk aplikasi analisis data.

Big Data Analytics adalah Alat dan teknik analisis yang akan sangat membantu dalam memahami big data dengan syarat algoritma yang menjadi bagian dari alat-alat ini harus mampu bekerja dengan jumlah besar pada kondisi real-time dan pada data yang berbeda-beda.

Bidang Pekerjaan baru Big Data Analytics:

1. Data Savvy Professionals: Seseorang yang tahu bagaimana untuk berpikir tentang data, bagaimana mengajukan jenis pertanyaan yang tepat sesuai dengan kebutuhan lembaga atau perusahaan atau lainnya dan mampu memahami dan mengklarifikasi jawaban hasil analisis yang mereka terima.
2. Technology and data enablers: Seseorang dapat memberikan dukungan integrasi antara data dengan teknologi yang sesuai, dan yang paling berkembang saat ini.
3. Data scientists (Memiliki bakat analitik yang mendalam/ Ilmuwan Data): orang-orang yang memiliki latar belakang yang kuat dalam algoritma-algoritma sistem cerdas, atau matematika terapan, atau ekonomi, atau ilmu pengetahuan lainnya melalui inferensi data dan eksplorasi.

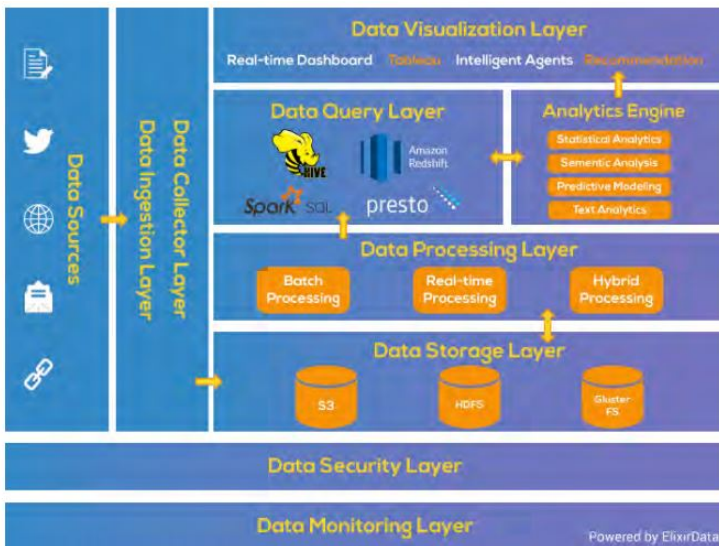
2.5 Arsitektur Big Data

Cara Terbaik untuk mendapatkan solusi dari Permasalahan Big Data (Big Data Solution) adalah dengan "Membagi Masalahnya". Big Data Solution dapat dipahami dengan baik menggunakan Layered Architecture. Arsitektur Layered dibagi ke dalam Lapisan yang berbeda dimana setiap lapisan memiliki spesifikasi dalam melakukan fungsi tertentu. Arsitektur tersebut membantu dalam merancang Data Pipeline (jalur data) dengan berbagai mode, baik Batch Processing System atau Stream Processing

System. Arsitektur ini terdiri dari 6 lapisan yang menjamin arus data yang optimal dan aman.

Data Ingestion Layer - Lapisan ini merupakan langkah awal untuk data yang berasal dari sumber tertentu dan akan memulai perjalanannya. Data disini akan dilakukan pemrioritasan dan pengkategorian, sehingga data dapat diproses dengan mudah diteruskan ke lapisan lebih lanjut. Tool yang dapat digunakan, yaitu Apache Flume, Apache Nifi (Pengumpulan dan Penggalian Data dari Twitter menggunakan Apache NiFi untuk Membangun Data Lake), Elastic Logstash. Data masuk ke dalam Data Lake dalam bentuk mentah, dan semua datanya disimpan, tidak hanya data yang digunakan saja, tapi juga data yang mungkin digunakan di masa depan. Di Data Lake semua data tersimpan dalam bentuk aslinya.

Lapisan Kolektor Data (Data Collector Layer) - Di Lapisan ini, lebih fokus pada penyaluran data dari lapisan penyerapan atau pengambilan data awal (ingestion) ke jalur data yang lainnya. Pada Lapisan ini, data akan dipisahkan sesuai dengan kelompoknya atau komponen-komponennya (Topik: kategori yang ditentukan pengguna yang pesannya dipublikasikan, Produser - Produsen yang memposting pesan dalam satu topik atau lebih, Konsumen berlangganan topik dan memproses pesan yang diposkan, Brokers - Pialang yang tekun dalam mengelola dan replikasi data pesan) sehingga proses analitik bisa dimulai. Tool yang dapat digunakan, yaitu Apache Kafka.



Gambar 2.8 Arsitektur Big Data

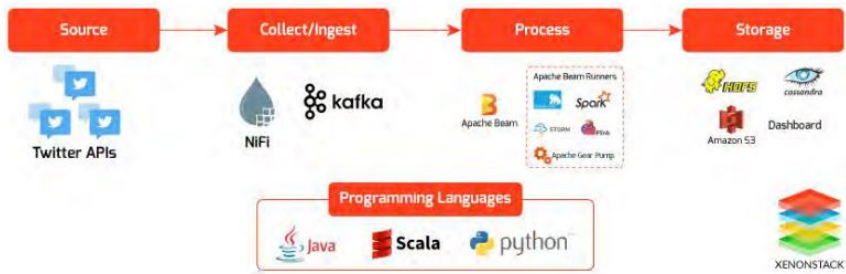
Lapisan Pengolahan Data (Data Processing Layer) - fokus utama lapisan ini adalah untuk sistem pemrosesan data pipeline atau dapat kita katakan bahwa data yang telah kita kumpulkan di lapisan sebelumnya akan diproses di lapisan ini. Di sini kita melakukan ekstraksi dan juga learning dari data untuk diarahkan ke tujuan yang

bermacam-macam, mengklasifikasikan arus data yang seharusnya diarahkan dan ini adalah titik awal di mana analitik telah dilakukan. Data pipeline merupakan komponen utama dari Integrasi Data. Data pipeline mengalirkan dan mengubah data real-time ke layanan yang memerlukannya, mengotomatiskan pergerakan dan transformasi data, mengolah data yang berjalan di dalam aplikasi Anda, dan mentransformasikan semua data yang masuk ke dalam format standar sehingga bisa digunakan untuk analisis dan visualisasi. Jadi, Data pipeline adalah rangkaian langkah yang ditempuh oleh data Anda. Output dari satu langkah dalam proses menjadi input berikutnya. Langkah-langkah dari Data pipeline dapat mencakup pembersihan, transformasi, penggabungan, pemodelan dan banyak lagi, dalam bentuk kombinasi apapun. Tool yang dapat digunakan, yaitu Apache Sqoop, Apache Storm, Apache Spark, Apache Flink.

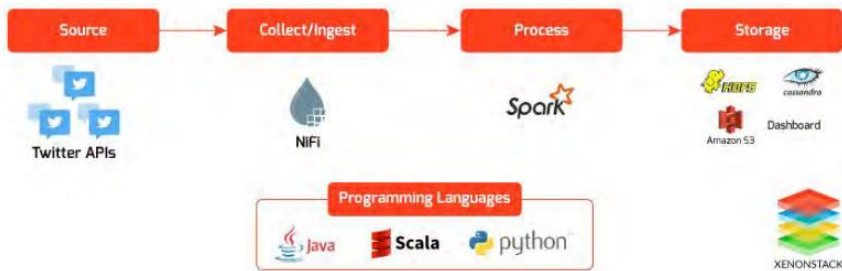
Lapisan Penyimpanan Data (Data Storage Layer) - Media penyimpanan menjadi tantangan utama, saat ukuran data yang digunakan menjadi sangat besar. Lapisan ini berfokus pada "tempat menyimpan data yang begitu besar secara efisien". Tool yang dapat digunakan, yaitu Apache Hadoop (HDFS), Gluster file systems (GFS), Amazon S3.

Lapisan Query Data (Data Query Layer) - lapisan ini merupakan tempat berlangsungnya pemrosesan secara analitik yang sedang dalam keadaan aktif. Di sini, fokus utamanya adalah mengumpulkan data value sehingga dapat dibuat lebih bermanfaat dan mudah digunakan untuk lapisan berikutnya. Tool yang dapat digunakan, yaitu Apache Hive, Apache (Spark SQL), Amazon Redshift, Presto.

Lapisan Visualisasi Data (Data Visualization Layer) - Proses Visualisasi, atau tahapan merepresentasikan data dalam bentuk visual, kemungkinan ini adalah tingkat yang paling bergengsi, di mana pengguna data pipeline dapat merasakan hasil laporan yang mendetail dan mudah dipahami dari data value yang telah divisualisasikan. Kita membutuhkan sesuatu yang akan menarik perhatian orang dari visualisasi data, sehingga membuat temuan Anda mudah dipahami dengan baik oleh mereka melalui visualisasi tersebut. Tool yang dapat digunakan, yaitu Tableau, Kibana sebagai Real-Time Dashboards, Angular.js sebagai Intelligence Agents misalnya agen dapat mengingat hal-hal yang mungkin Anda sudah lupa, dengan cerdas meringkas data yang kompleks, belajar dari Anda dan bahkan membuat rekomendasi untuk Anda, menemukan dan memfilter informasi saat Anda melihat data perusahaan atau berselancar di Internet dan tidak tahu di mana informasi yang tepat, React.js sebagai sistem recommender untuk memprediksi tentang kriteria pengguna, yaitu menentukan model penggunaanya seperti apa.



Gambar 2.9 Data Integration Using Apache NiFi dan Apache Kafka



Gambar 2.10 Integrating Apache Spark dan NiFi for Data Lakes

Apache Spark digunakan secara luas untuk pengolahan Big Data. Spark bisa mengolah data di kedua mode yaitu Pengolahan Batch Mode dan Streaming Mode. Apache NiFi ke Apache Spark melakukan transmisi data menggunakan komunikasi situs ke situs. Dan output port-nya digunakan untuk mempublikasikan data dari sumbernya (source). Apache Spark adalah mesin pemrosesan data dalam memori, yang cepat dan ringkas dengan mode pengembangan API yang elegan dan ekspresif, yang memungkinkan pengguna melakukan proses secara streaming, menggunakan pembelajaran mesin (machine learning), atau SQL yang memerlukan akses berulang-ulang secara cepat terhadap kumpulan data. Dengan Spark yang berjalan di Apache Hadoop YARN, developer sekarang dapat membuat aplikasi dengan memanfaatkan kehandalan dari Spark, untuk memperoleh wawasan, dan memperkaya data sains mereka dengan memasukkan data dalam satu kumpulan data besar di Hadoop.

2.6 Key Roles Kunci Sukses Proyek Analitik

1. Bussines User

Business User: Seseorang yang memahami wilayah domain (kondisi existing) dan dapat mengambil manfaat besar dari hasil proyek analitik, dengan cara konsultasi dan menyarankan tim proyek pada scope proyek, hasil, dan operasional output (terkait dengan cara mengukur suatu variabel). Biasanya yang memenuhi peran ini adalah analis bisnis, manajer lini, atau ahli dalam hal pokok yang mendalam.

2. Project Sponsor

Project Sponsor: Bertanggung jawab terkait asal proyek. Memberi dorongan, persyaratan proyek dan mendefinisikan masalah core bisnis. Umumnya menyediakan dana dan konsep pengukur tingkat nilai output akhir dari tim kerja. Menetapkan prioritas proyek dan menjelaskan output yang diinginkan.

3. Project Manager

Project Manager: Memastikan bahwa pencapaian utama proyek dan tujuan terpenuhi tepat waktu dan sesuai dengan kualitas yang diharapkan.

4. Business Intelligence Analyst

Business Intelligence Analyst: Menyediakan keahlian dalam domain bisnis berdasarkan pemahaman yang mendalam mengenai data, indikator kinerja utama (KPI), metrik kunci, dan intelijen bisnis dari perspektif pelaporan. Analis Business Intelligence umumnya membuat dashboard (panel kontrol) dan laporan dan memiliki pengetahuan tentang sumber data dan mekanismenya.

5. Database Administrator (DBA)

Database Administrator (DBA): Set up dan mengkonfigurasi database untuk mendukung kebutuhan analitik. Tanggung jawab ini mungkin termasuk menyediakan akses ke database keys atau tabel dan memastikan tingkat keamanan yang sesuai berada di tempat yang berkaitan dengan penyimpanan data.

6. Data Engineer

Data Engineer: Memiliki keterampilan teknis yang mendalam untuk membantu penyetelan query SQL untuk pengelolaan data dan ekstraksi data, dan mendukung untuk konsumsi data ke dalam sandbox analitik. Data Engineer melakukan ekstraksi data aktual dan melakukan manipulasi data yang cukup besar untuk memfasilitasi kebutuhan proyek analitik. Insinyur data (Data Engineer) bekerja sama dengan ilmuwan data (Data Scientist) untuk membantu membentuk data yang sesuai dengan cara yang tepat untuk analisis .

7. Data Scientist

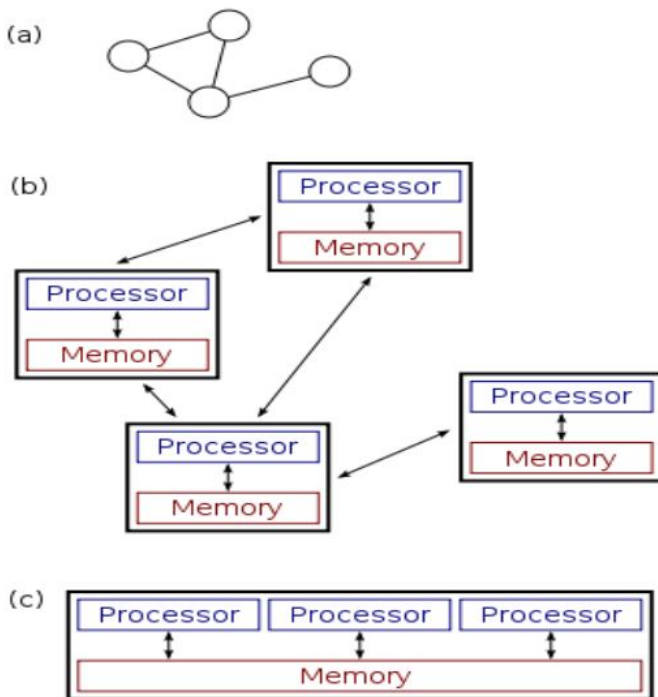
Data Scientist (Ilmuan Data): Menyediakan keahlian untuk teknik analitis, pemodelan data, dan menerapkan teknik analitis yang valid untuk masalah yang

diberikan. Memastikan melalui keseluruhan analitik tujuannya dapat terpenuhi. Merancang dan mengeksekusi metode analitis dan melakukan pendekatan lainnya dengan data yang tersedia untuk proyek tersebut.

2.7 Konsep Pengolahan Big Data

Bayangkan ada sekumpulan data yang sangat besar (Big Data), bagaimana kita dapat melakukan Parallel atau Distributed Processing.

File Sistem Terdistribusi (Distributed File System, atau disingkat dengan DFS) adalah file sistem yang mendukung sharing files dan resources dalam bentuk penyimpanan persistent (tetap) untuk tujuan tertentu di sebuah network.



Gambar 2.11 Distributed System (a) dan Paralel System (b)

2.8 Introduction to Hadoop

Hadoop: Suatu software framework (kerangka kerja perangkat lunak) open source berbasis Java di bawah lisensi Apache untuk aplikasi komputasi data besar secara intensif.

Hadoop File System dikembangkan menggunakan desain sistem file yang terdistribusi. Tidak seperti sistem terdistribusi, HDFS sangat faulttolerant dan dirancang menggunakan hardware low-cost. Atau dalam arti lain, Hadoop adalah Software platform (platform perangkat lunak) sebagai analytic engine yang memungkinkan seseorang dengan mudah untuk melakukan pembuatan penulisan perintah (write) dan menjalankan (run) aplikasi yang memproses data dalam jumlah besar, dan di dalamnya terdiri dari: HDFS Hadoop Distributed File System dan MapReduce of-line computing engine.

Dalam komputasi, platform menggambarkan semacam (hardware architecture) arsitektur perangkat keras atau (software framework) kerangka kerja perangkat lunak (termasuk kerangka kerja aplikasi), yang memungkinkan perangkat lunak dapat berjalan.

Ciri khas dari platform meliputi arsitekturnya komputer, sistem operasi, bahasa pemrograman dan runtime libraries atau GUI yang terkait.

2.9 Hadoop Distributed File System (HDFS)

Hadoop terdiri dari HDFS (Hadoop Distributed file System) dan Map Reduce. HDFS sebagai direktori di komputer dimana data hadoop disimpan. Untuk pertama kalinya, direktori ini akan di format agar dapat bekerja sesuai spesifikasi dari Hadoop. HDFS sebagai file system, tidak sejajar dengan jenis file system dari OS seperti NTFS, FAT32. HDFS ini menumpang diatas file system milik OS baik Linux, Mac atau Windows.

Data di Hadoop disimpan dalam cluster. Cluster biasanya terdiri dari banyak node atau komputer/server. Setiap node di dalam cluster ini harus terinstall Hadoop untuk bisa jalan.

Hadoop versi 1.x ada beberapa jenis node di dalam cluster:

- **Name Node:** Ini adalah node utama yang mengatur penempatan data di cluster, menerima job dan program untuk melakukan pengolahan dan analisis data misal melalui Map Reduce. Name Node menyimpan metadata tempat data di cluster dan juga replikasi data tersebut.
- **Data Node:** Ini adalah node tempat data ditempatkan. Satu block di HDFS/data node adalah 64 MB. Jadi sebaiknya data yang disimpan di HDFS ukurannya minimal 64 MB untuk memaksimalkan kapasitas penyimpanan di HDFS.
- **Secondary Name Node:** Bertugas untuk menyimpan informasi penyimpanan data dan pengolahan data yang ada di name node. Fungsinya jika name node mati dan diganti dengan name node baru maka name node baru bisa langsung bekerja dengan mengambil data dari secondary name node.
- **Checkpoint Node dan Backup Node:** Checkpoint node melakukan pengecekan setiap interval waktu tertentu dan mengambil data dari name node. Dengan check point node maka semua operasi perubahan pada data terekam. Namun,

secondary name node hanya perlu menyimpan check point terakhir. Backup Node juga berfungsi sama, hanya bedanya data perubahan yang disimpan dilakukan di memory bukan di file seperti checkpoint dan secondary node.

Kelemahan HDFS di hadoop versi 1.x adalah jika name node mati. Maka seluruh cluster tidak bisa digunakan sampai name node baru dipasang di cluster.

Hadoop versi 2.x ada beberapa jenis node di dalam cluster:

- Lebih dari satu name nodes. Hal ini berfungsi sebagai implementasi dari High Availability. Hanya ada satu name node yang berjalan di cluster (aktif) sedangkan yang lain dalam kondisi pasif. Jika name node yang aktif mati/rusak, maka name node yang pasif langsung menjadi aktif dan mengambil alih tugas sebagai name node.
- Secondary name node, checkpoint node dan backup node tidak lagi diperlukan. Meskipun ketiga jenis node tersebut menjadi optional, tetapi kebanyakan tidak lagi ada di cluster yang memakai hadoop versi 2.x. Hal ini karena selain fungsi yang redundan, juga lebih baik mengalokasikan node untuk membuat tambahan name node sehingga tingkat High Availability lebih tinggi.
- Data node tidak ada perubahan yang signifikan di versi hadoop 2.x dari versi sebelumnya.

Meskipun konteks yang kita bicarakan disini adalah dalam cluster, Hadoop juga bisa dijalankan dalam single node. Dalam single node maka semua peran diatas berada dalam satu komputer. Biasanya single node ini digunakan hanya untuk training atau development. Bukan untuk produksi skala enterprise.

BAB 3

GOOGLE COLAB

3.1 *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 `'''`.

3.2 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

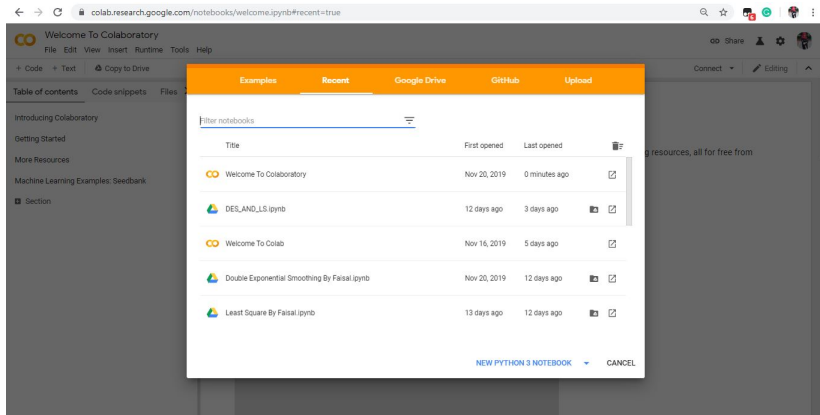
Dapat dengan mudah menghubungkan *Google Colab* dengan jupyter notebook di komputer dengan *local runtime*, menghubungkan dengan *Google Drive*, atau dengan Github.

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.

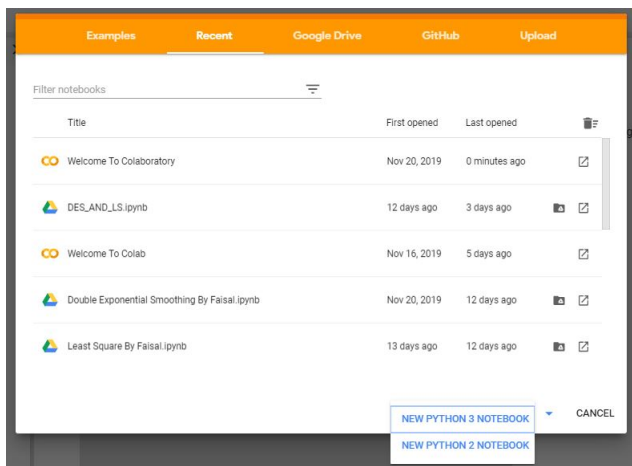
3.3 Cara Penggunaan *Google Colab*

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



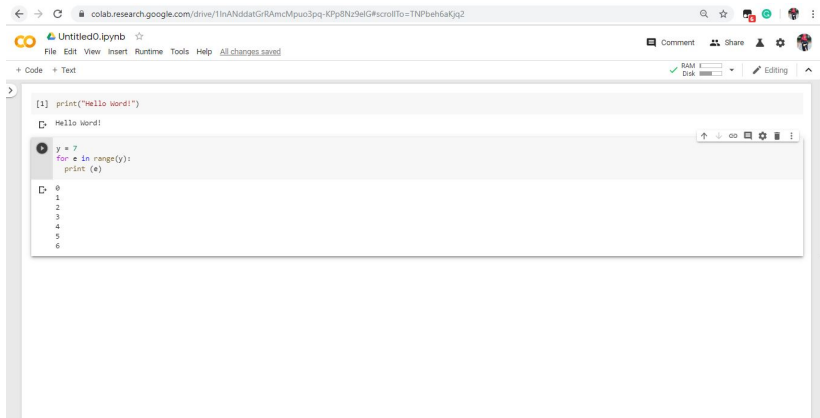
Gambar 3.1 Tampilan Awal *Google Colab*

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



Gambar 3.2 Tampilan Membuat Notebook Baru

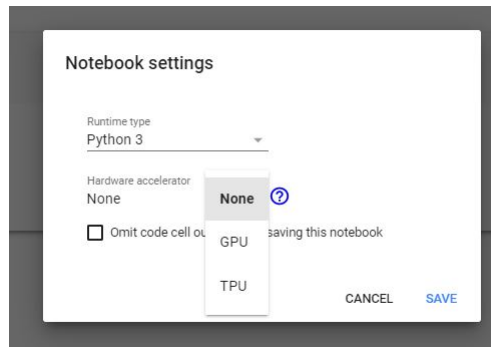
- Setelah itu akan menampilkan tampilan ke halaman yang mirip dengan Jupyter Notebook. Nantinya, setiap notebook yang kita buat akan disimpan di *Google Drive*.



Gambar 3.3 Tampilan *Google Colab*

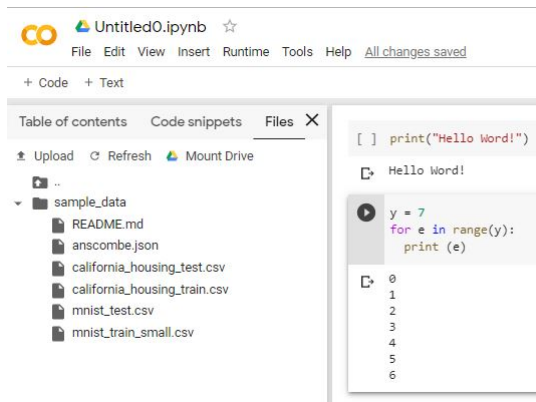
4. Pengaturan GPU

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

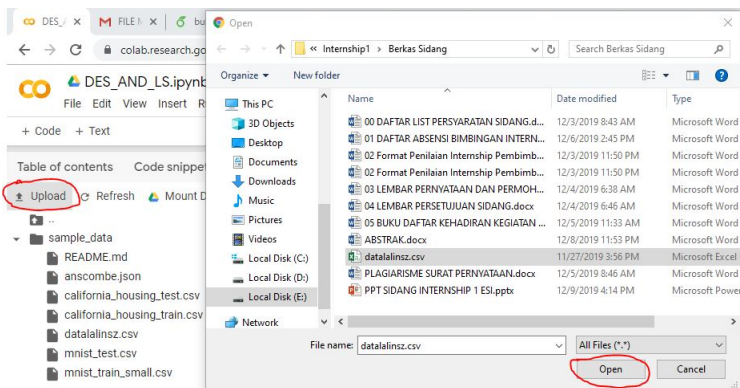


Gambar 3.4 Tampilan Pengaturan *Hardware Accelerator*

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.

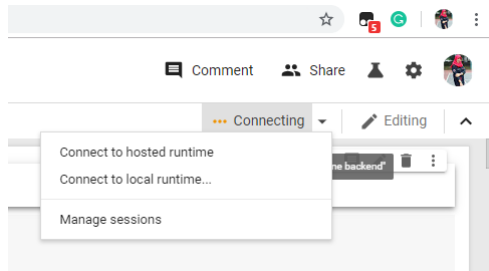


Gambar 3.5 Tampilan File Data csv di *Goole Colab*

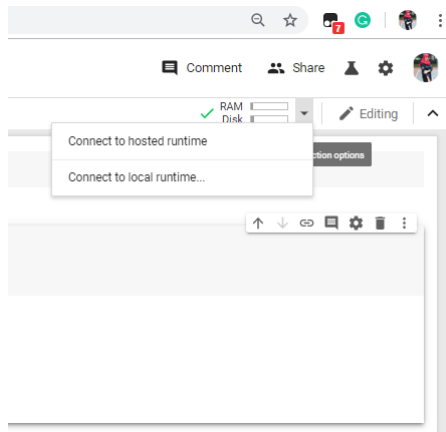


Gambar 3.6 Tampilan Upload Data csv

6. Ketika kita membuat file baru di *Google Colab* tentu belum langsung terkoneksi ke komputing di google seperti pada gambar 3.7. Lalu setelah itu pilih *Connect to hosted runtime*. Maka setelah itu akan terkoneksi seperti gambar 3.8.

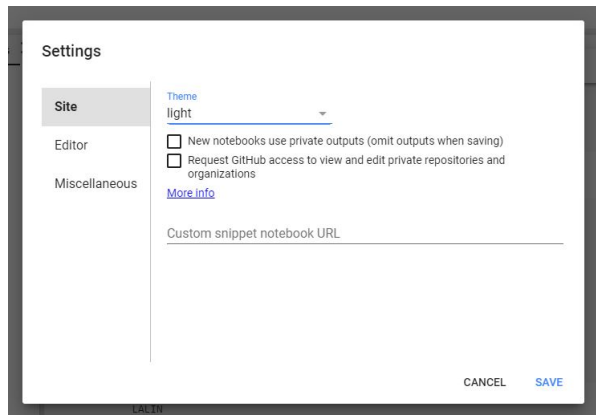


Gambar 3.7 Tampilan Sebelum Konek

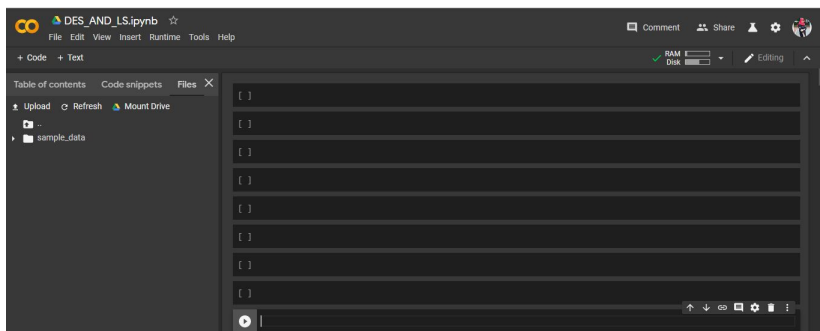


Gambar 3.8 Tampilan Setelah Konek

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



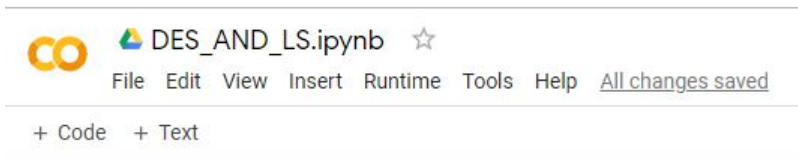
Gambar 3.9 Tampilan Pengaturan Notebook



Gambar 3.10 Tampilan Berhasil Merubah Warna

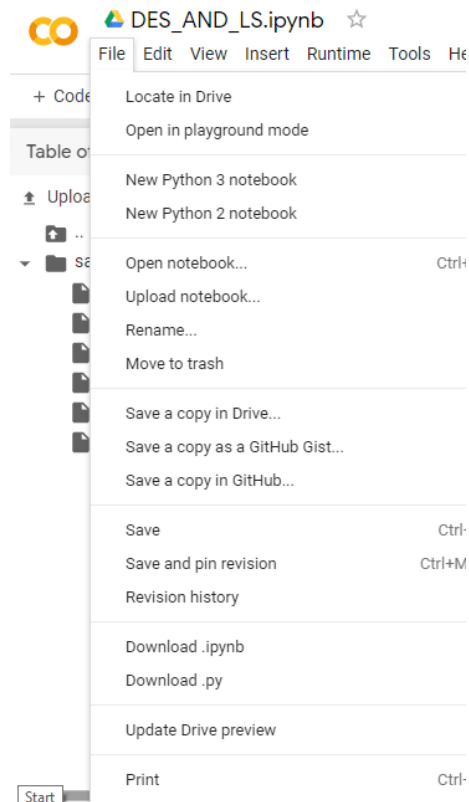
3.4 Tools *Google Colab*

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*.



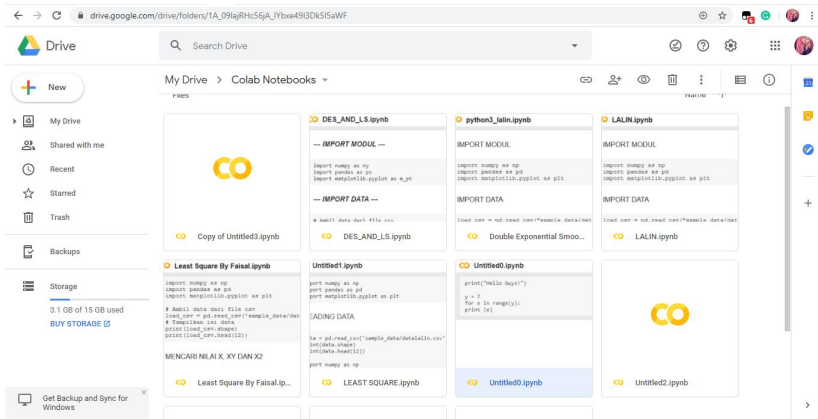
Gambar 3.11 Tools *Google Colab*

1. File :



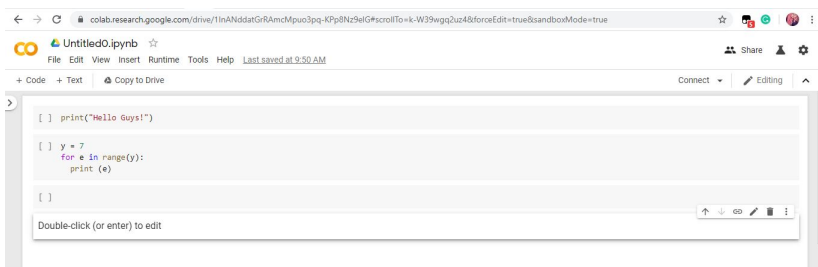
Gambar 3.12 Tools *File*

- **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 3.13 *Locate in Drive*

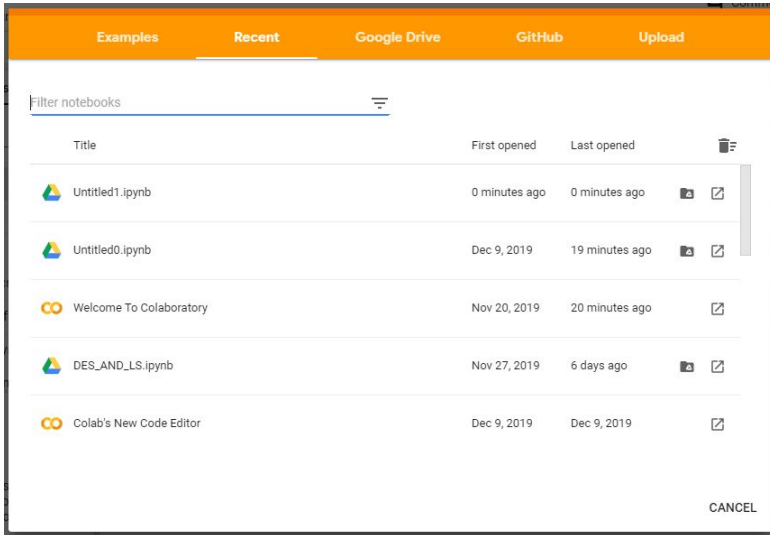
- **Open in playground mode** : Tidak terdapat *output* yang disimpan dalam mode tersebut, tetapi tidak dapat menggunakan fungsi *tools rename* dan *save and pin revision*.



Gambar 3.14 *Open in playground mode*

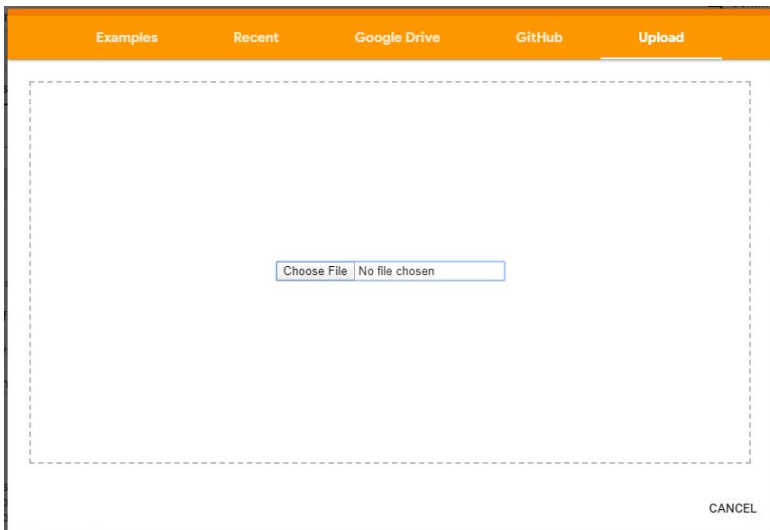
- **New Python 3 notebook** : Membuat file baru di tab baru dengan format Python 3 *notebook*.
- **New Python 2 notebook** : Membuat file baru dengan di tab baru dengan format Python 2 *notebook*.

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



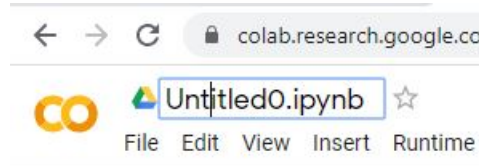
Gambar 3.15 *Open notebook*

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



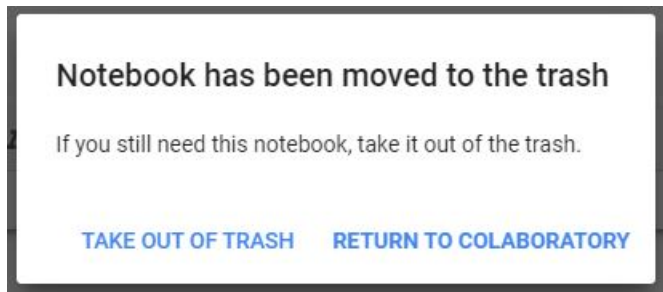
Gambar 3.16 *Upload notebook*

- **Rename** : Untuk mengubah nama dari file Python sesuai yang di inginkan, setelah itu akan diarahkan seperti gambar 3.17.



Gambar 3.17 *Rename*

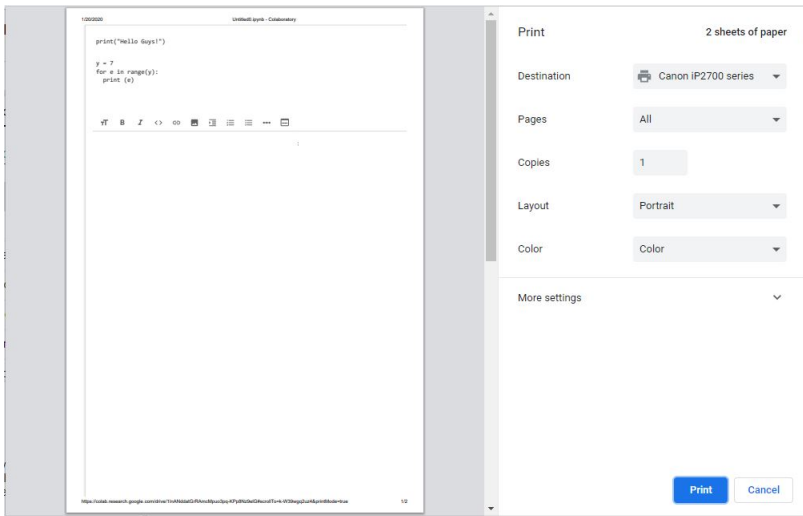
- **Move to trash** : Untuk menghapus file yang sedang di buka dan terdapat dua pilhan yaitu dikeluarkan dari tempat sampah atau kembali ke kolaboratori.



Gambar 3.18 *Move to trash*

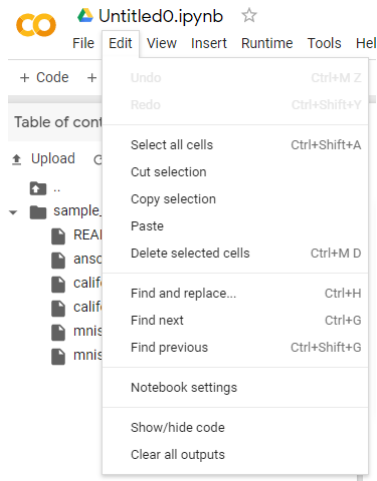
- **Save a copy in Drive** : Untuk menyimpan file *notebook* yang sudah dibuat kedalam *Google Drive*.
- **Save a copy as a GitHub Gist** : Untuk menyimpan file *notebook* yang sudah dibuat kedalam bentuk *Github Gist*.
- **Save a copy in GitHub** : Untuk menyimpan file *notebook* yang sudah dibuat kedalam *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 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*.



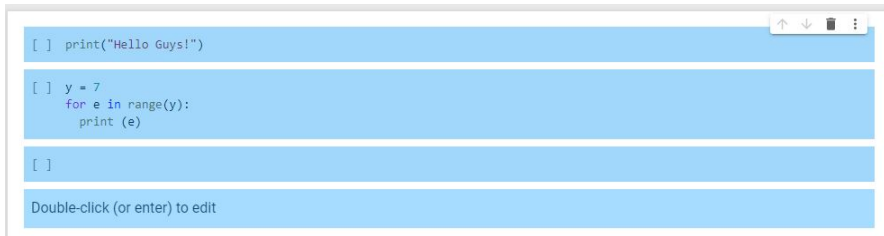
Gambar 3.19 *Print*

2. Edit :



Gambar 3.20 *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 3.21 *Select all cells*

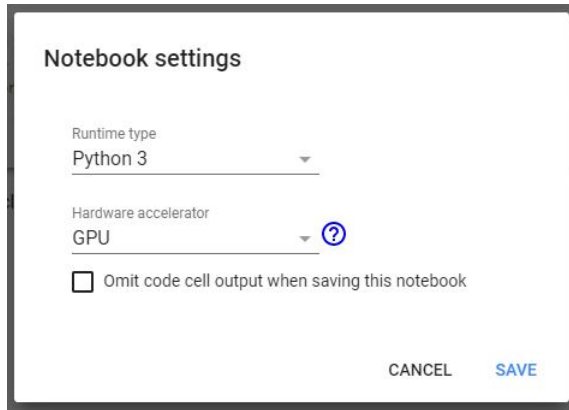
- **Cut selection** : Berfungsi untuk mengcut atau memotong *source code* yang sudah di pilih yang ada di dalam *notebook Google Colab*.
- **Copy selection** : Berguna untuk mencopy kata-kata atau *source code* yang sudah terseleksi atau terpilih.
- **Paste** : Yang digunakan untuk menempelkan kata, 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 mengganti kata tersebut sesuai dengan keinginan. Akan tampil pada bagian pojok kanan bawah pada google colab.



Gambar 3.22 *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*.

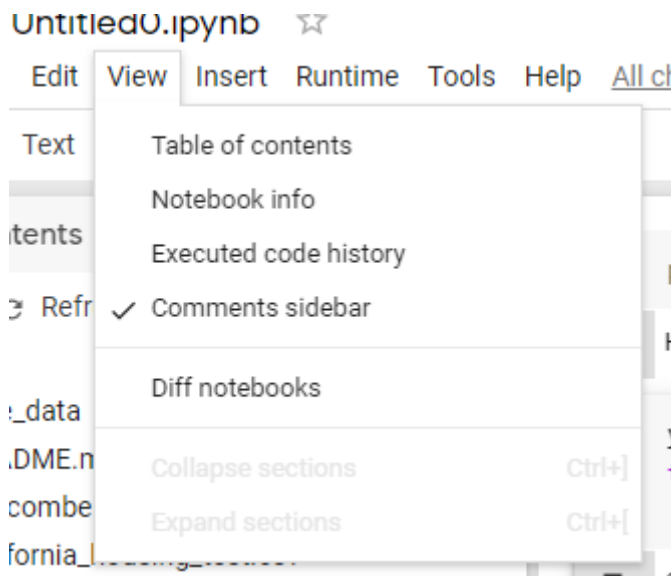
- *Notebook settings* : Untuk mengatur pengaturan yang ada di dalam *notebook* seperti mengatur runtime type, hardware accelerator.



Gambar 3.23 *Notebook settings*

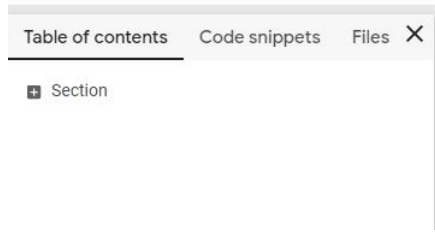
- *Show/hide code* : Berfungsi untuk menampilkan dan menyembunyikan *code*.
- *Clear all outputs* : Berfungsi untuk menghapus semua *output* yang ada.

3. View :



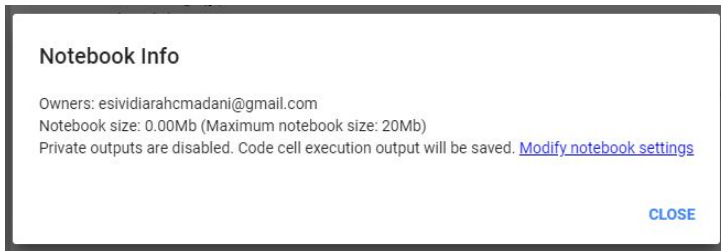
Gambar 3.24 *Tools View*

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



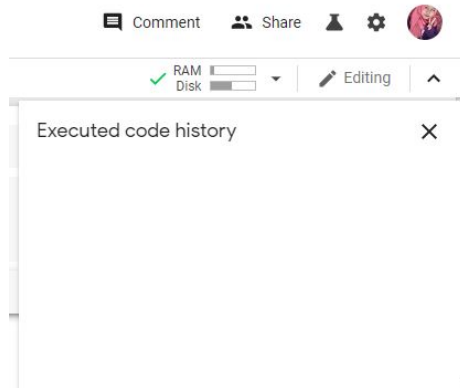
Gambar 3.25 *Table of contents*

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



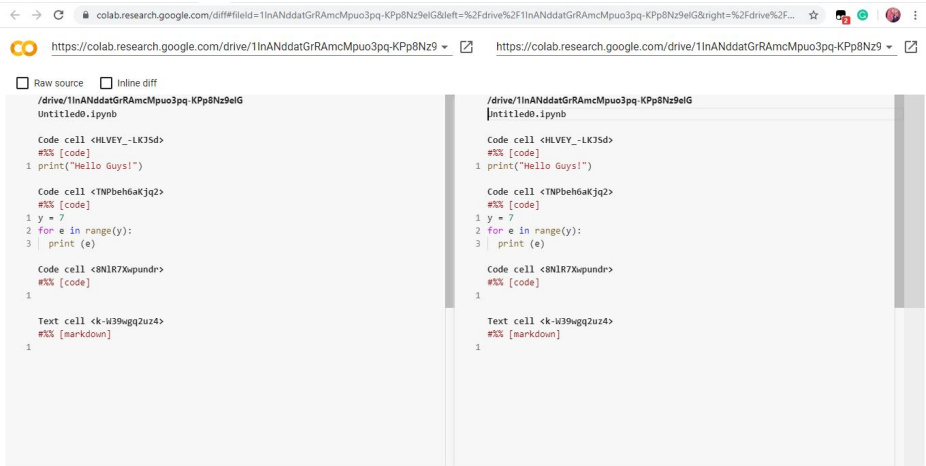
Gambar 3.26 *Notebook info*

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

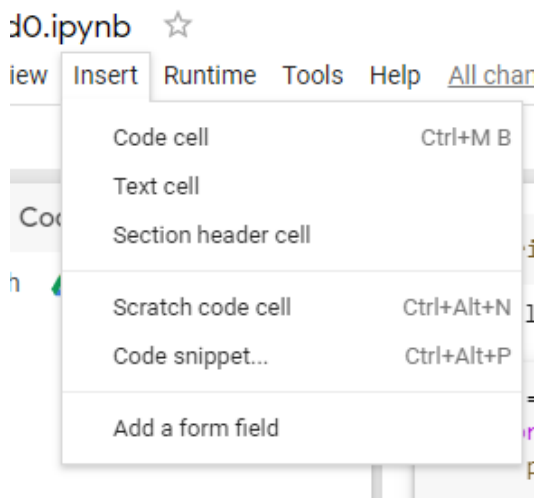


Gambar 3.27 *Executed code history*

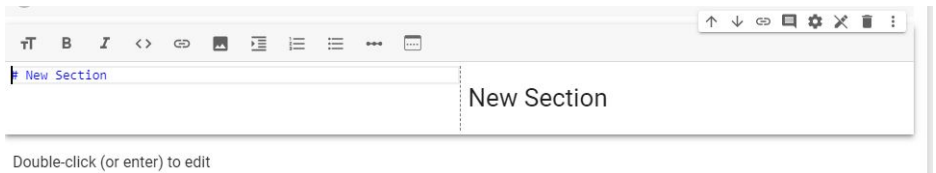
- *Comments sidebar* : Untuk menambahkan komentar di dalam *source code* di sampingnya.
- *Diff notebooks* : Perintah atau menu yang berfungsi untuk menampilkan *notebook* yang baru atau yang berbeda.

Gambar 3.28 *Diff notebooks*

4. Insert :

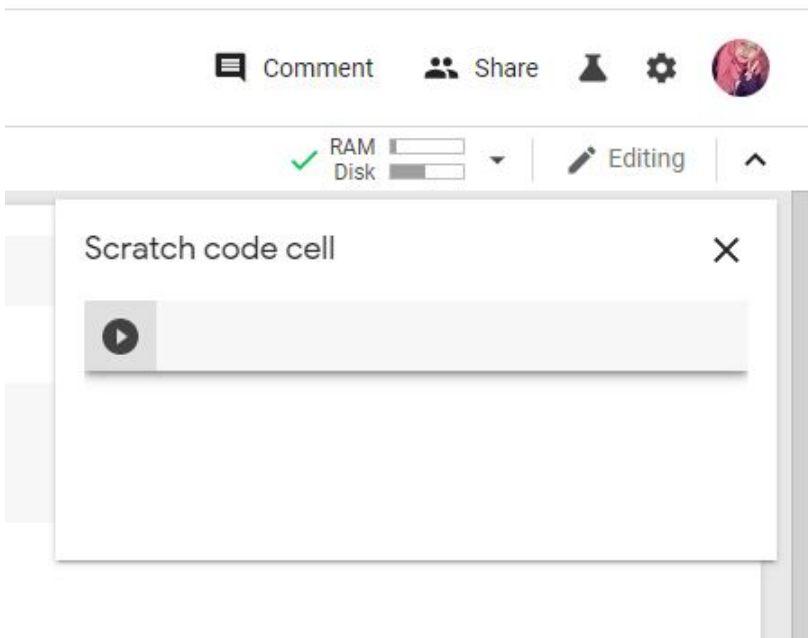
Gambar 3.29 *Tools Insert*

- *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.



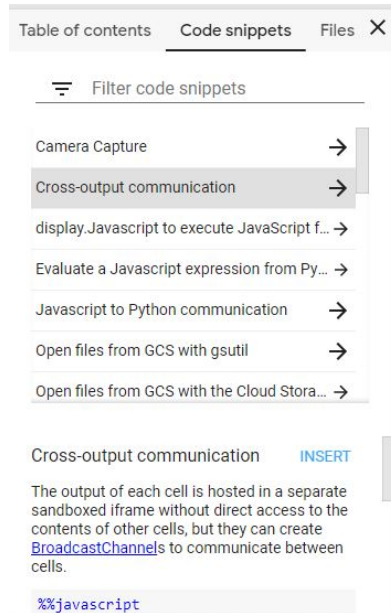
Gambar 3.30 *Section header cell*

- *Scratch code cell* : untuk menggoreskan kode sel atau membuat dan merung-ing *script* di dalam *Scratch code cell* seperti pada 3.31.



Gambar 3.31 *Scratch code cell*

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

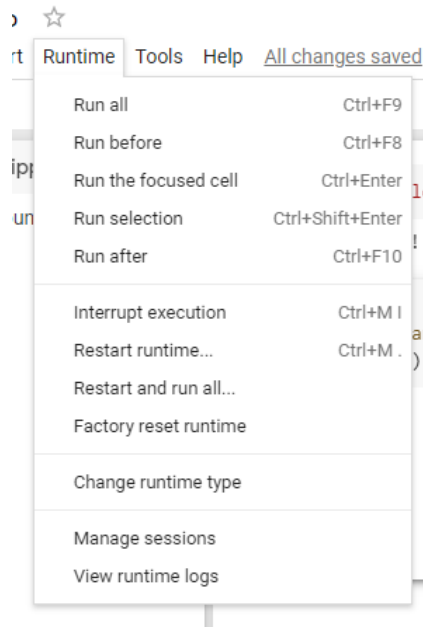


Gambar 3.32 *Code snippet*

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

Gambar 3.33 *Add a from field*

5. Runtime :



Gambar 3.34 *Tools Runtime*

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



Gambar 3.35 *Run all*

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



Gambar 3.36 *Run before*

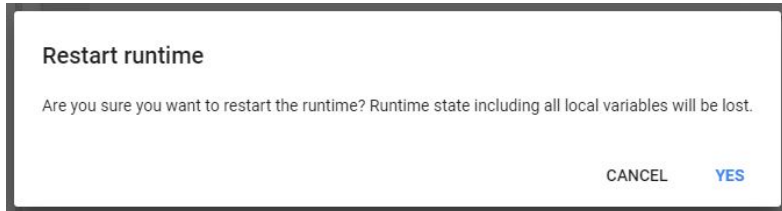
- *Run the focused cell* : Perintah yang hanya akan meruning jika satu cell yang di fokuskan.



Gambar 3.37 *Run the focused cell*

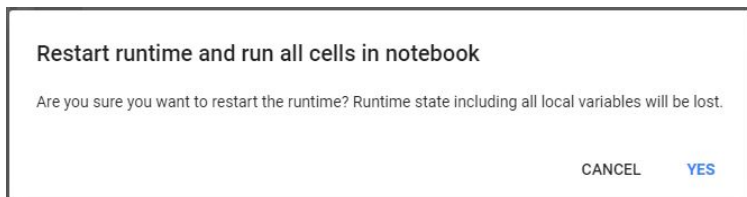
- *Run selection* : Perintah untuk menjalankan *source code* yang di seleksi saja.
- *Run after* : Perintah untuk menjalankan pada *source code* sebelumnya.

- *Interrupt execution* : Perintah untuk menjalankan eksekusi interupsi yang sudah di perintah.
- *Restart runtime* : Perintah yang berfungsi atau berguna untuk memulai kembali menjalankan *source code*.



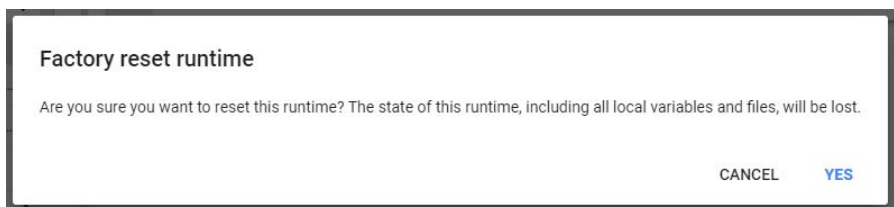
Gambar 3.38 *Popup Restart runtime*

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



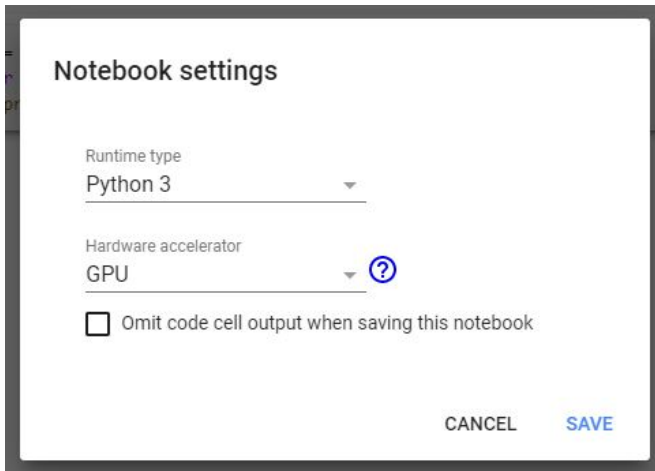
Gambar 3.39 *Popup Restart and run all*

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



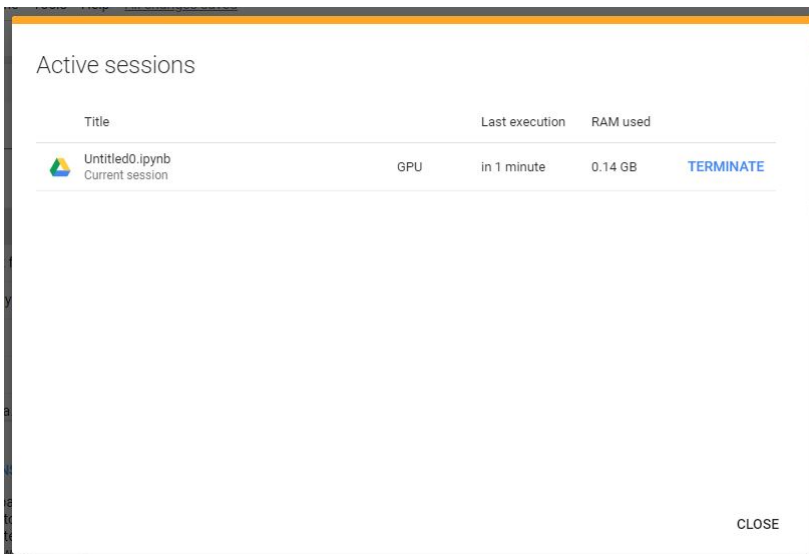
Gambar 3.40 *Popup Factory reset runtime*

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

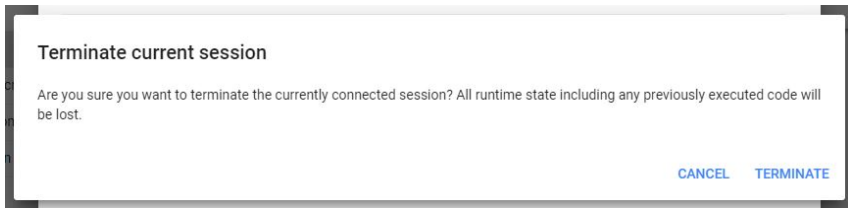


Gambar 3.41 *Change runtime type*

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



Gambar 3.42 *Manage sessions*



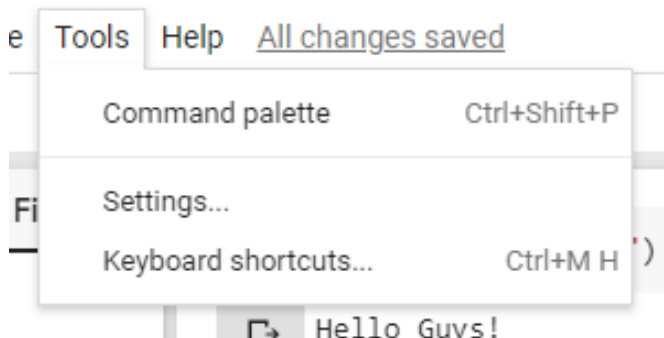
Gambar 3.43 Setelah Terminate

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



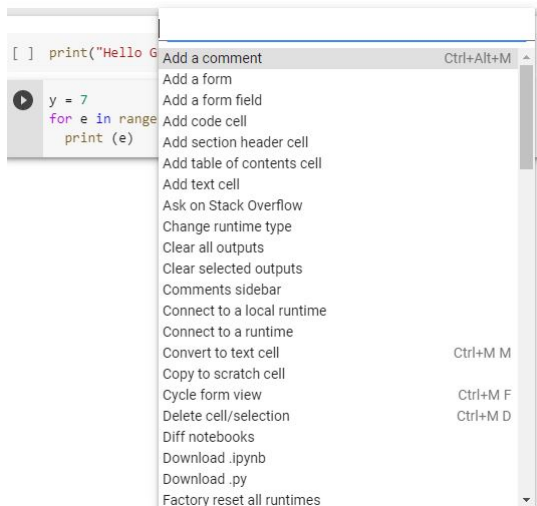
Gambar 3.44 View runtime logs

6. Tools :



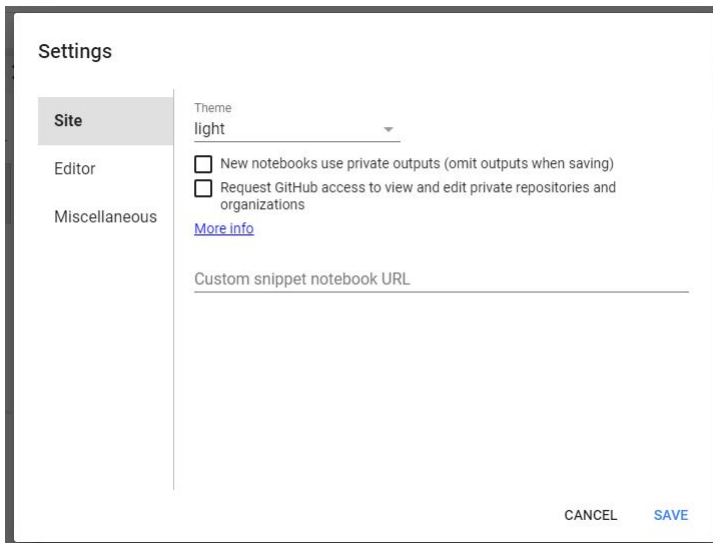
Gambar 3.45 Tools

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



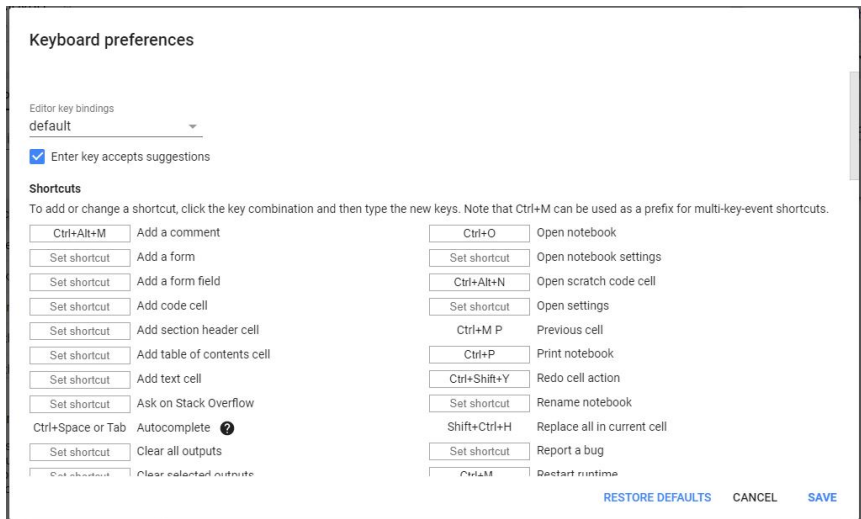
Gambar 3.46 *Command palette*

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



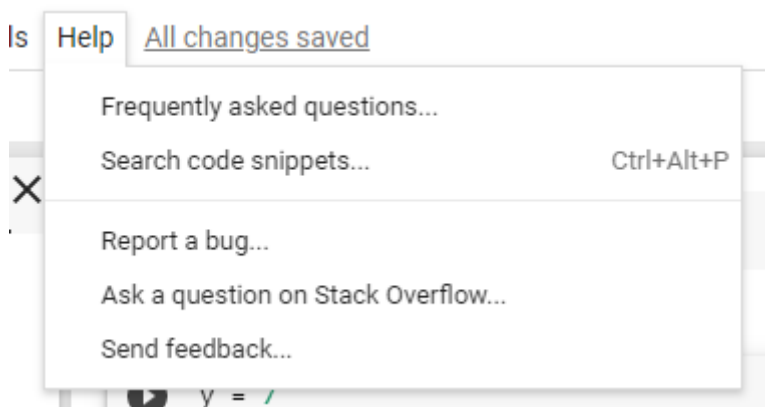
Gambar 3.47 *Settings*

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



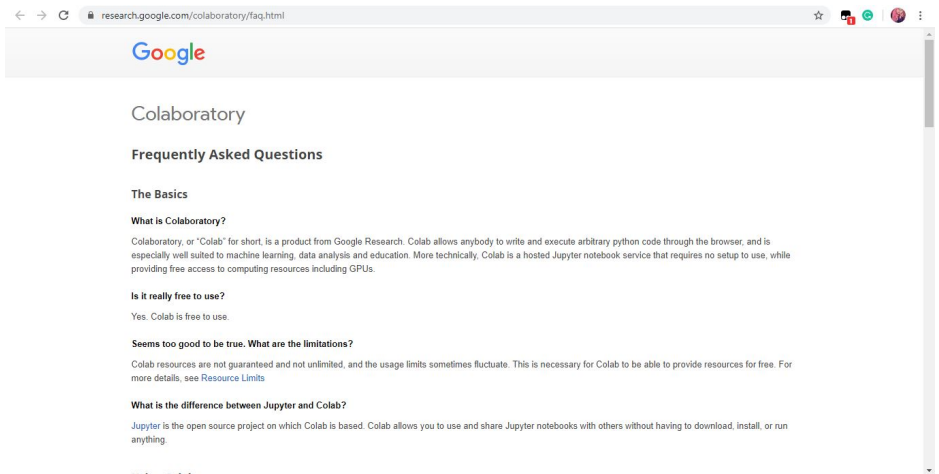
Gambar 3.48 *Keyboard shortcuts*

7. Help :



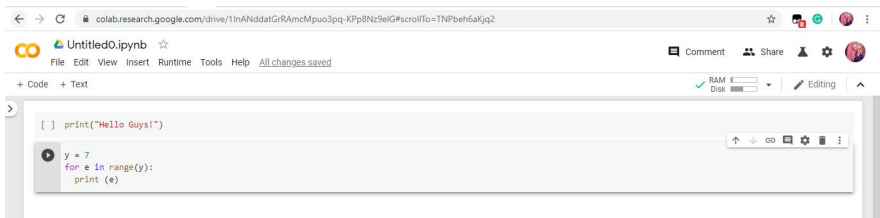
Gambar 3.49 *Tools Help*

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



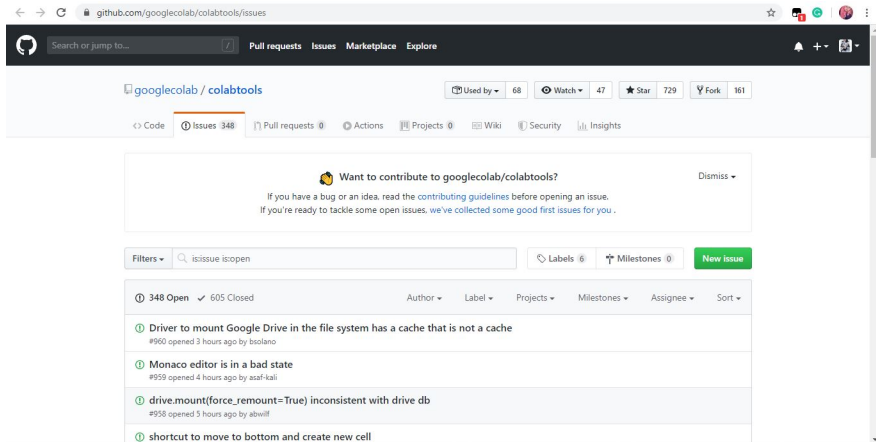
Gambar 3.50 *Frequently Asked Questions*

- *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 3.51.



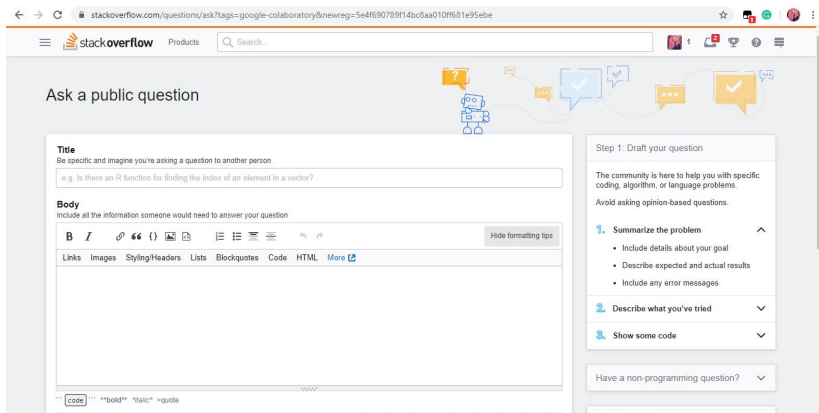
Gambar 3.51 *Search code snippets*

- **Report a bug** : Perintah untuk melaporkan bug yang ada pada google colab yang setelah itu akan muncul pada tampilan github seperti pada gambar 3.52.



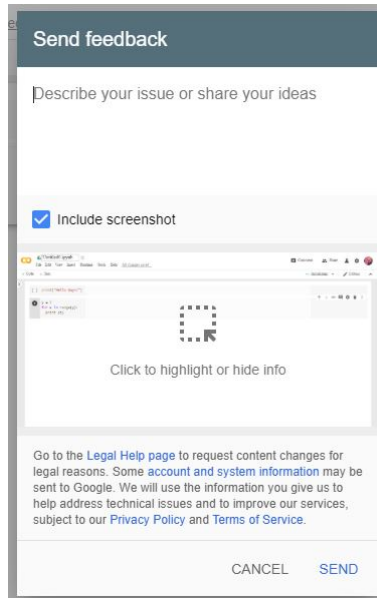
Gambar 3.52 Report a bug

- **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.



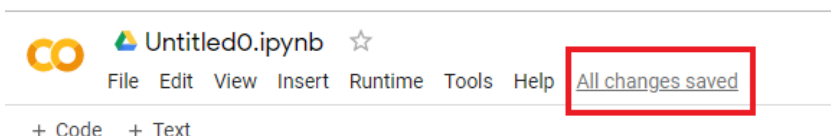
Gambar 3.53 Ask a question on Stack Overflow

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



Gambar 3.54 *Send feedback*

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

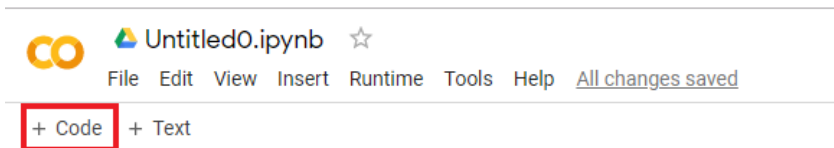


Gambar 3.55 *All changes saved*

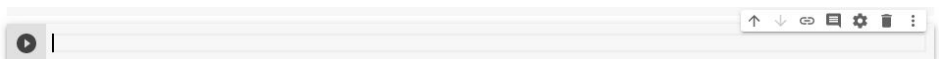


Gambar 3.56 *Tampilan All changes saved*

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

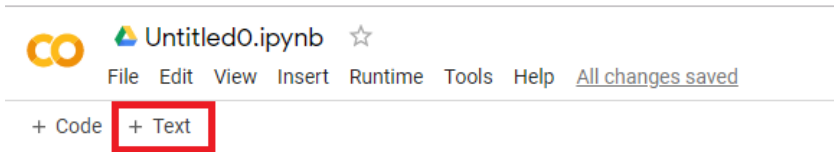


Gambar 3.57 *Code*



Gambar 3.58 *Tampilan Code*

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



Gambar 3.59 *Text*



Gambar 3.60 *Tampilan Text*

BAB 4

LIBRARY PYTHON

4.1 Konsep Penting untuk Dimengerti tentang Library Python

Sebelum masuk ke penjelasan berbagai macam library Python, kita harus memahami beberapa konsep. Misalnya, deep learning (pembelajaran secara dalam) adalah sebuah proses dari machine learning (pembelajaran mesin). Apakah kamu tahu bagaimana orang dapat belajar dari kesalahan mereka? Hal yang sama berlaku untuk komputer. Deep learning bertujuan untuk membuat mesin belajar melalui contoh.

Istilah lain yang relevan adalah neural network atau jaringan saraf, yang menyerupai otak manusia. Dengan cara bagaimana? Neural network adalah kombinasi dari algoritma yang bertujuan untuk meniru cara manusia yang mampu mengidentifikasi berbagai pola. Oleh karena itu, konsep ini mengambil biologi manusia dan menerapkannya ke dunia pemrograman untuk memperkenalkan pengenalan gambar dan ucapan (hanya salah satu opsi).

Pertama, kamu harus memahami bahwa library untuk Python tidak jauh berbeda dari perpustakaan biasa yang kamu kunjungi untuk mencari dan memilih buku yang menarik. Keduanya merupakan kumpulan sumber informasi.

Python libraries

Namun, alih-alih buku, kamu memilih modul yang akan kamu terapkan selama proses coding kamu. Semua developer profesional memanfaatkan modul yang ditulis dengan baik. Jika ada cara yang mudah untuk melakukan sesuatu, mengapa kamu tidak mengambil cara ini?

Ketika kamu mulai meneliti library Python, kamu akan dihujani oleh sejumlah library asli maupun library pihak ketiga. Ada banyak koleksi modul yang tersedia. Karena itu, kamu mungkin akan merasa bingung ketika harus memutuskan yang mana untuk dijelajahi. Jika kamu adalah seorang programmer yang ingin menjadi unggul dalam beberapa domain yang berbeda, mungkin akan sulit untuk memilih library yang paling cocok.

Kamu sebaiknya sudah tahu bahwa Python adalah bahasa yang sangat fleksibel. Ia adalah permata di dunia pemrograman karena penggunaannya bervariasi dari ilmu data/data science, pengembangan web, dan bahkan pembelajaran mesin. Kalau kamu adalah seorang programmer Python pemula, kami menganjurkan untuk mengambil kursus ini untuk memperdalam pengetahuan kamu.

Secara keseluruhan, berbagai library Python mencakup modul untuk area tertentu. Siapkah kita untuk memulai ekspedisi untuk mencari tahu apa itu TensorFlow, PyTorch, Numpy, Sklearn, dan perpustakaan populer lainnya?

Namun sebelumnya, apakah kamu kesulitan mencari pekerjaan sebagai programmer Python? Dalam kasus seperti itu, kami sangat menyarankan untuk membaca beberapa pertanyaan dan jawaban wawancara Python yang biasanya ditanyakan oleh para pengusaha dan perusahaan. Jika kamu tidak bisa menjawabnya, kamu mungkin tampaknya tidak siap untuk posisi tersebut. Katakanlah salah satu pertanyaan wawancara Python mengharuskan kamu untuk berbicara tentang library Python.

Banyaknya kelebihan pada pemrograman python seperti efisiensi, keterbacaan kode dan kecepatan telah membuat python menjadi bahasa pemrograman yang banyak digunakan oleh para data scientist. Python menjadi pilihan untuk para data scientist dan machine learning engineer untuk mengembangkan model dan berbagai aplikasi terkait data science.

Karena penggunaannya yang luas, Python memiliki banyak library yang memudahkan para ilmuwan data / data scientist untuk menyelesaikan tugas-tugas rumit tanpa banyak gangguan pengkodean. Berikut adalah 3 library Python yang paling banyak digunakan untuk data science.

4.2 Library Terbaik untuk Dipertimbangkan

PI adalah singkatan untuk application programming interface (antarmuka pemrograman aplikasi). Ia membuka jendela untuk interaksi antara aplikasi melalui komunikasi mesin-ke-mesin. Python memiliki framework (kerangka kerja) yang mempercepat proses pembuatan API. Oleh karena itu, misi kita adalah membahas secara singkat library paling umum untuk Python yang dapat kamu pilih:

1. Flask adalah framework web yang berkembang pesat, dirancang untuk proses perancangan API yang lebih efisien. Ini hanya salah satu dari banyak kemungkinan penggunaan Flask. Secara umum, ini adalah framework untuk pengem-

bankan aplikasi web. Flash ringan, menawarkan dukungan untuk pengujian unit dan mengamankan cookie untuk sesi di sisi klien. Para developer memuji framework ini karena didokumentasikan dengan baik, artinya kamu akan menemukan banyak contoh penggunaan untuk dipelajari.

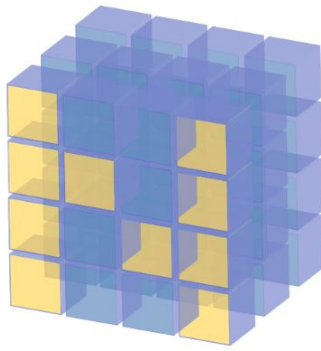
2. Django adalah salah satu framework web pihak ketiga berbasis Python. Di antara library Python lainnya, tujuan utama Django dari framework ini adalah untuk menyederhanakan proses pengembangan situs web kompleks yang digerakkan oleh database. Perpustakaan Django menyediakan banyak alat (tool) manajemen. Oleh karena itu, para developer akan dapat menghasilkan bagian-bagian kode tanpa harus beralih ke alat lain. Django REST adalah framework untuk membuat API Web dengan kode yang minimal.
3. Falcon adalah framework web yang ringan dan sesuai dengan SWGI, yang dirancang untuk membangun RESTful APIs. Para pemula menghargai tutorial yang terdokumentasi dengan baik yang menyediakan banyak panduan untuk pembuatan proyek pertama. Falcon berjalan pada hardware apa saja dan hanya bergantung pada dua dependensi pihak ketiga.
4. Eve adalah framework REST API berbasis Python gratis, ditenagai oleh Flask dan Cerberus. Ia memungkinkan layanan web RESTful yang unik dan kaya fitur untuk dikembangkan secara cepat. Framework ini mendukung MongoDB dan sangat kompatibel karena ekstensi.

4.3 library NumPy

NumPy (kependekan dari Numerical Python) adalah salah satu library teratas yang dilengkapi dengan sumber daya yang berguna untuk membantu para data scientist mengubah Python menjadi alat analisis dan pemodelan ilmiah yang kuat. Library Open source terpopuler ini tersedia di bawah lisensi BSD. Ini adalah pustaka Python dasar untuk melakukan tugas dalam komputasi ilmiah. NumPy adalah bagian dari ekosistem berbasis Python yang lebih besar dari tool open source yang disebut SciPy.

Perpustakaan memberdayakan Python dengan struktur data substansial untuk mudah melakukan perhitungan multi-dimensi (multi-dimensional arrays) dan perhitungan matrik. Selain penggunaannya dalam menyelesaikan persamaan aljabar linier (linear algebra equations) dan perhitungan matematis lainnya, NumPy juga digunakan sebagai wadah multi-dimensi serbaguna untuk berbagai jenis data generik.

Lebih hebatnya, NumPy terintegrasi dengan bahasa pemrograman lain seperti C / C++ dan Fortran. Fleksibilitas perpustakaan NumPy memungkinkannya untuk dengan mudah dan cepat bergabung dengan berbagai database dan tools. Sebagai contoh, mari kita lihat bagaimana NumPy (disingkat np) dapat digunakan untuk menganalisis dua matriks.



NumPy

Gambar 4.1 NumPy

List pada Python tidak mendukung penuh kemudahan scientific computing, sebagai contoh kita akan lakukan operasi penjumlahan pada 2 list.

```
In [1]: 1 # list pada python
        2 a = [1, 2, 3]
        3 b = [4, 5, 6]
        4 a + b

Out[1]: [1, 2, 3, 4, 5, 6]
```

```
In [2]: 1 # penjumlahan dilakukan iterasi
        2 result = []
        3 for first, second in zip(a, b):
        4     result.append(first + second)
        5 result

Out[2]: [5, 7, 9]
```

Gambar 4.2

Ketika kita ingin menjumlahkan tiap elemen pada list a dan list b, hasilnya dengan operator + adalah penggabungan (concat) keduanya. Tentu tidak sesuai yang diharapkan, maka kita harus menggunakan perulangan for untuk menambahkan tiap elemen pada list a dan list b. Proses penjumlahan list yang menggunakan perulangan for membutuhkan waktu yang lama dan tidak efisien dari sisi penulisan code.

4.3.1 Pengenalan NumPy Arrays

Membuat Array

Lakukan import terlebih dahulu library numpy as np. Penggunaan as disini, artinya kita menggantikan pemanggilan numpy dengan prefix np untuk proses berikutnya.

```
In [3]: 1 import numpy as np

In [4]: 1 # membuat array
        2 a = np.array([1, 2, 3])
        3 a

Out[4]: array([1, 2, 3])
```

Gambar 4.3

Untuk membuat sebuah array, kita menggunakan fungsi `array()` yang terdapat pada NumPy. Pada NumPy, terdapat `upcasting`, yaitu ketika tipe data element array tidak sama, dilakukan penyamaan tipe data pada yang lebih tinggi. Misalkan kita membuat array numeric dengan semua element bertipe integer, kecuali 1 element bertipe float, maka otomatis akan dilakukan `upcasting` menjadi tipe float pada semua element array.

Untuk melakukan pengecekan tipe pada array menggunakan fungsi `type()`.

NumPy array merupakan sebuah objek `ndarray`, yang merupakan singkatan dari `n-dimensional array`. Tipe Data untuk Element Pengecekan tipe data element pada array menggunakan fungsi `dtype`.

Dalam membuat sebuah array, kita dapat menetapkan tipe data dengan menambahkan parameter `dtype`.

Array `a` memiliki tipe data `int32` dan `int64` yang keduanya sama-sama bertipekan integer. Perbedaan keduanya pada kapasitas penyimpanan data. Pada `int32` mampu menampung hingga `(-2,147,483,648 to +2,147,483,647)` sedangkan `int64` mampu menampung hingga `(-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807)`. Penting bagi kita memperhatikan tipe data beserta kapasitas penyimpanannya agar dapat mengalokasikan memory penyimpanan dengan baik. Beberapa tipe data standar yang terdapat pada NumPy adalah sebagai berikut:

NumPy array memiliki keunggulan mendukung operasi pada data dimensional seperti Vektor dan Matriks. Untuk mengetahui jumlah dimensi pada data menggunakan fungsi `ndim`.

Array `a` memiliki jumlah dimensi 1. Jika kita membentuk array dengan 2-dimensi, maka jumlah dimensinya adalah 2, begitu juga dengan dimensi yang lebih besar. Array Shape Pada fungsi `shape` menghasilkan sebuah tuple yang berisikan panjang sebuah array pada tiap dimensi.

Array `a` memiliki shape 3, yaitu panjang array pada 1-dimensi array. Operasi pada Array NumPy memudahkan kita untuk operasi `elementwise` pada Vektor dan Matriks seperti penjumlahan, perkalian, pangkat, dan operasi lainnya.

Pada NumPy telah disediakan `universal functions (ufunc)` yaitu fungsi yang dapat melakukan operasi pada NumPy array dengan eksekusi elemen-demi-elemen. `ufunc` merupakan sebuah `vectorized wrapper` untuk sebuah fungsi yang memiliki input dan output yang spesifik. Contoh dari `ufunc` misalkan fungsi `sin` dan `cos`.

Element pada Array Array Indexing Indeks pada suatu array dimulai pada indeks ke-0. Untuk mengakses element pada array, kita menggunakan indeks sebagai alamat elemen pada array.

Kita dapat melakukan assign nilai baru pada suatu element berdasarkan alamat indeks. Maka, setelah dilakukan assign nilai element pada indeks ke-0 berganti menjadi 10. Perlu kita perhatikan jika dtype pada array adalah integer, misalkan int32. Kemudian kita assign nilai baru pada salah satu elemen dengan tipe data float. Maka, nilai baru tersebut akan dipaksa menjadi tipe int32 sesuai dengan tipe data pada array.

Multi-Dimensional Arrays NumPy array memudahkan kita untuk membuat array multi-dimensi.

Fungsi shape pada array multi-dimensi menghasilkan tuple berisikan (jumlah baris, jumlah kolom).

Untuk mengetahui jumlah elemen pada array multi-dimensi menggunakan fungsi size.

Untuk mengetahui jumlah dimensi dari array multi-dimensi menggunakan fungsi ndim. Jumlah dimensi pada array a adalah 2, karena merupakan array 2-dimensi.

Pada array multi-dimensi kita dapat mengakses nilai elemen berdasarkan indeks dengan pemisah tanda koma [,]. Misalkan kita ingin mendapatkan data pada baris ke-1 dan kolom ke-3 dari array a, maka dilakukan perintah a[1, 3].

Misalkan, kita akan mengganti data pada baris ke-1 dan kolom ke-3 pada array a dengan nilai -1, kita dapat langsung assign nilai baru tersebut pada indeks.

Slicing Selain kita dapat mengambil sebuah element array dengan cara indexing seperti cara di atas, kita dapat melakukan slicing, yaitu melakukan ekstraksi elemen atau beberapa elemen pada array, dengan menggunakan tanda [:]. var[lower:upper:step] Perlu kita perhatikan, elemen pada lower akan dimasukkan hasil slicing, sedangkan element pada upper TIDAK dimasukkan hasil slicing, dan step adalah jarak antar elemen.

Kita dapat mendefinisikan indeks dengan nilai negatif. Misalkan indeksnya adalah [-1], maka indeks terdapat pada elemen ke-1 yang dari lokasi akhir elemen suatu array. NumPy masih menyediakan banyak sekali fungsi yang sangat memudahkan kita untuk scientific computing. Untuk mempelajari lebih jauh dapat mengakses dokumentasi NumPy.

4.4 Pandas

Pandas adalah library hebat lain yang dapat meningkatkan keterampilan Python Anda untuk data science. Sama seperti NumPy, Pandas milik keluarga perangkat lunak open source SciPy dan tersedia di bawah lisensi perangkat lunak bebas BSD.

Pandas menawarkan alat serbaguna dan kuat untuk struktur data dan melakukan analisis data yang luas. Library ini berfungsi dengan baik dengan data dunia nyata yang tidak lengkap, tidak terstruktur, dan tidak teratur dan dilengkapi dengan tool untuk membentuk, menggabungkan, menganalisis, dan memvisualisasikan datasets.

Pandas (Python for Data Analysis) adalah library Python yang fokus untuk proses analisis data seperti manipulasi data, persiapan data, dan pembersihan data. Pan-

das menyediakan struktur data dan fungsi high-level untuk membuat pekerjaan dengan data terstruktur/tabular lebih cepat, mudah, dan ekspresif. Dalam pandas terdapat dua objek yang akan dibahas pada tutorial ini, yaitu DataFrame dan Series. DataFrame adalah objek yang memiliki struktur data tabular, berorientasi pada kolom dengan label baris dan kolom. Sedangkan Series adalah objek array 1-dimensi yang memiliki label.

Pandas memadukan library NumPy yang memiliki kemampuan manipulasi data yang fleksibel dengan database relasional (seperti SQL). Sehingga memudahkan kita untuk melakukan reshape, slice dan dice, agregasi data, dan mengakses subset dari data. Menurut penulis buku Python for Data Analysis sekaligus pembuat pandas, Wes McKinney, nama pandas berdasarkan dari panel data, yaitu istilah ekonometrik untuk data multi-dimensi terstruktur, dan berdasarkan dari kata yang merupakan fungsional library itu sendiri yaitu Python data analysis.

Ada tiga jenis struktur data di library ini:

- Series: single-dimensional, array homogen
- DataFrame: two-dimensional dengan kolom yang diketik secara heterogen
- Panel: three-dimensional, array size-mutable



Gambar 4.4 Pandas

Series adalah objek 1-dimensi yang berisi sequence nilai dan berasosiasi dengan label data, yang disebut indeks. Untuk membuat sebuah Series, kita dapat membentuknya dari sebuah array dengan memanggil fungsi Series pada pandas.

Dapat kita lihat, pada Series obj ditampilkan dua buah kolom, bagian kiri adalah index dan bagian kanan adalah values. Pada index, karena kita tidak mendefinisikan index pada data, maka secara default dimulai dari integer 0 hingga N-1 (dimana N adalah panjang data). Kita dapat mendefinisikan index dengan menambahkan parameter index saat membuat objek Series.

Kita dapat mengambil index dari objek Series dengan menggunakan fungsi index dan mengambil values dari objek Series dengan menggunakan fungsi values.

Untuk mengakses suatu value pada objek Series, kita dapat menggunakan index sebagai alamat value tersebut. Hal ini memungkinkan untuk melakukan assign nilai baru pada objek Series.

Hasil operasi aritmatika tambah pada 2 objek Series mirip dengan operasi join pada pengolahan database. Indeks yang tidak memiliki kesamaan pada 2 objek Series akan memiliki value NaN.

DataFrame merupakan tabel data yang terdapat kolom dan baris, dimana nilai-nilai yang terdapat di dalamnya dapat berupa tipe berbeda seperti numeric, string, boolean, dll. DataFrame mirip dengan data 2-dimensi dengan adanya baris dan kolom. Selain itu, DataFrame bisa dikatakan gabungan dari dictionary objek Series yang memiliki indeks yang sama. Terdapat berbagai macam cara untuk membentuk objek DataFrame. Salah satu cara yang biasa dilakukan untuk membentuk objek DataFrame dengan menggunakan data masukan berupa dictionary.

Kita dapat menggunakan fungsi `shape` untuk mengetahui jumlah baris dan kolom dari DataFrame. Fungsi `shape` pada DataFrame memiliki hasil keluaran yang sama dengan fungsi `shape` pada NumPy, dimana menghasilkan keluaran sebuah tuple dari jumlah baris dan jumlah kolom.

Fungsi `info()` sangat berguna untuk mengetahui keterangan dari objek DataFrame yang kita buat seperti index dari DataFrame lengkap dengan range dari index, jumlah kolom beserta informasi tiap kolom untuk null data dan tipe data, dan jumlah total penggunaan memory pada tiap kolom dalam satuan bytes. Saat kita input data dictionary pada DataFrame, kita tidak perlu mendefinisikan tipe data untuk masing-masing kolom, karena secara otomatis pandas akan memberikan tipe data sesuai dengan values untuk tiap kolom, meskipun kita juga bisa mendefinisikan tipe data secara manual.

Misalkan kita memiliki objek DataFrame yang memiliki baris hingga jutaan, kita tidak ingin menampilkan data secara keseluruhan karena akan menghabiskan memory. Kita dapat menggunakan fungsi `head()` dan `tail()` untuk menampilkan data secara default untuk 5 data teratas dan 5 data terbawah.

Seperti pada Series, kita juga dapat mengakses kolom pada DataFrame menggunakan fungsi `columns` dan untuk mengakses indeks dari DataFrame menggunakan fungsi `index`. Jika ingin mendapatkan data pada DataFrame secara keseluruhan menggunakan fungsi `values` yang akan menghasilkan output berupa array 2-dimensi sesuai dengan jumlah baris dan kolom.

DataFrame menyediakan fungsi `describe()` untuk mengetahui statistika data untuk data numeric seperti count, mean, standard deviation, maximum, minum, dan quartile. Untuk data string, misalkan data tersebut adalah kategori, kita dapat menggunakan fungsi `value_counts()` untuk mengetahui jumlah tiap kategori pada data.

Mengakses Data pada DataFrame

Terkadang kita butuh mengakses kolom tertentu atau lebih spesifik elemen tertentu pada DataFrame. Untuk mengakses suatu data, alamat data tersebut adalah pada nama kolom sebagai petunjuk lokasi kolom dan indeks sebagai petunjuk lokasi baris. Misalkan untuk mengakses semua data pada kolom populasi, maka kita menggunakan perintah `frame[populasi]`. Perlu diingat karena nama kolom adalah tipe data string, maka kita menggunakan petik.

Sedangkan mengakses data pada baris tertentu, kita menggunakan fungsi `loc[indeks]`. Kita juga bisa mendapatkan lebih dari 1 baris dengan menggunakan titik dua `:`, misalkan kita ingin mengakses indeks 23, maka menggunakan perintah `loc[2:3]`.

Elemen pada DataFrame dapat diakses dengan mendefinisikan nama kolom dan indeks baris secara bersamaan. Misalkan kita ingin mendapatkan data populasi pada indeks ke-2, maka digunakan perintah `frame[populasi][2]`.

Pandas masih menyediakan banyak sekali fungsi yang sangat memudahkan kita untuk analisis data seperti mengisi data kosong dengan fungsi `fillna()`, menerapkan suatu fungsi pada DataFrame menggunakan `apply()`, mengubah format DataFrame dari wide ke long menggunakan `melt()`, dan masih banyak lagi. Untuk mempelajari lebih jauh dapat mengakses dokumentasi pandas.

4.5 Matplotlib

Matplotlib juga merupakan bagian dari paket inti SciPy dan ditawarkan di bawah lisensi BSD. Ini adalah library ilmiah Python populer yang digunakan untuk menghasilkan visualisasi yang sederhana dan kuat. Anda dapat menggunakan kerangka kerja Python untuk ilmu data untuk menghasilkan grafik, chart, histogram, dan bentuk dan gambar lain yang kreatif tanpa perlu khawatir menulis banyak baris kode. Sebagai contoh, mari kita lihat bagaimana perpustakaan Matplotlib dapat digunakan untuk membuat bar chart sederhana.








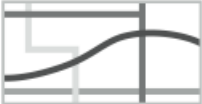






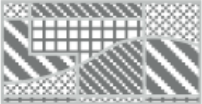


Gambar 4.5 Matplotlib

Matplotlib adalah library Python yang fokus pada visualisasi data seperti membuat plot grafik. Matplotlib pertama kali diciptakan oleh John D. Hunter dan sekarang telah dikelola oleh tim developer yang besar. Awalnya matplotlib dirancang untuk menghasilkan plot grafik yang sesuai pada publikasi jurnal atau artikel ilmiah. Matplotlib dapat digunakan dalam skrip Python, Python dan IPython shell, server aplikasi web, dan beberapa toolkit graphical user interface (GUI) lainnya. Visualisasi dari matplotlib adalah sebuah gambar grafik yang terdapat satu sumbu atau lebih. Setiap sumbu memiliki sumbu horizontal (x) dan sumbu vertikal (y), dan data yang direpresentasikan menjadi warna dan glyphs seperti marker (contohnya bentuk lingkaran) atau lines (garis) atau poligon.

Gambar di bawah menunjukkan bagian-bagian dari visualisasi matplotlib dibuat oleh Nicolas P. Rougier.

Hal yang penting dalam visualisasi data adalah penentuan warna, tekstur, dan style yang menarik untuk dilihat dan representatif terhadap data. Seorang Cartographer yaitu Jacques Bertin mengembangkan rekomendasi berikut untuk pemilihan informasi visual yang cocok, dan kita dapat menerapkannya menggunakan matplotlib.

	<i>Points</i>	<i>Lines</i>	<i>Areas</i>	<i>Best to show</i>
<i>Shape</i>		<i>possible, but too weird to show</i>	<i>cartogram</i>	<i>qualitative differences</i>
<i>Size</i>			<i>cartogram</i>	<i>quantitative differences</i>
<i>Color Hue</i>				<i>qualitative differences</i>
<i>Color Value</i>				<i>quantitative differences</i>
<i>Color Intensity</i>				<i>qualitative differences</i>
<i>Texture</i>				<i>qualitative & quantitative differences</i>

Gambar 4.7 Pandas

Untuk memulai menggunakan matplotlib, lakukan import terlebih dahulu library `matplotlib.pyplot` as `plt`. Penggunaan `as` disini, artinya kita menggantikan pemanggilan fungsi `pyplot` pada `matplotlib` dengan prefix `plt` untuk proses berikutnya. Disini terdapat magic command `% matplotlib inline`, untuk pengaturan pada backend matplotlib agar setiap grafik ditampilkan secara inline, yaitu akan ditampilkan langsung pada cell notebook.

Line plot berguna untuk melacak perubahan pada periode waktu pendek dan panjang. Ketika terdapat perubahan kecil, line plot lebih baik dalam melakukan visualisasi dibandingkan grafik bar.

Tutorial kali ini akan membuat plot grafik line menggunakan gelombang cos. Kita akan menggunakan `numpy` untuk generate data gelombang cos dengan jumlah data 100 yang berjarak dari 0 sampai 2.

Sumbu x dan y pada kurva cos seharusnya memiliki rasio yang sama karena keduanya merupakan satuan unit yang sama. Kita dapat mengaturnya menggunakan fungsi `.set_aspect`.

Kita dapat mengatur bentuk marker menggunakan parameter `linestyle`, ukuran marker menggunakan parameter `markersize`, warna menggunakan parameter `color`, dan memberikan legend menggunakan parameter `legend`.

Scatter plot biasanya digunakan untuk melakukan observasi dan menunjukkan hubungan relasi antara dua variabel numeric. Titik-titik pada scatter plot juga dapat menggambarkan pola dari data secara keseluruhan. Matplotlib menyediakan fungsi `scatter()` untuk mempermudah dalam visualisasi scatter plot.

Bar plot digunakan untuk membandingkan perubahan tiap waktu pada beberapa kelompok data. Bar plot sangat bagus digunakan dalam visualisasi ketika perubahan data sangat besar dibandingkan dengan line plot. Bar plot biasanya memiliki dua sumbu yaitu sumbu x untuk jenis kelompok dan sumbu y untuk proporsi kelompok. Matplotlib menyediakan fungsi `bar()` untuk mempermudah dalam visualisasi bar plot.

Matplotlib masih menyediakan banyak sekali fungsi untuk melakukan visualisasi dengan berbagai jenis grafik seperti pie, histogram, grafik 3-dimensi, dll. Untuk mempelajari lebih jauh dapat mengakses dokumentasi matplotlib.

DAFTAR PUSTAKA

1. R. Awangga, "Sampeu: Servicing web map tile service over web map service to increase computation performance," in *IOP Conference Series: Earth and Environmental Science*, vol. 145, no. 1. IOP Publishing, 2018, p. 012057.