

APLIKASI MUSICROOT

APLIKASI MUSICROOT

**Muchamad Innal Kariem
Cecep Gunawan
Rolly M. Awangga**
Informatics Research Center



Kreatif Industri Nusantara

Penulis:

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:

M. Yusril Helmi Setyawan

Penyunting:

Syafrial Fachrie Pane

Khaera Tunnisia

Diana Asri Wijayanti

Desain sampul dan Tata letak:

Deza Martha Akbar

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasisih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara
apapun tanpa ijin tertulis dari penerbit

*'Jika Kamu tidak dapat
menahan lelahnya
belajar, Maka kamu harus
sanggup menahan
perihnya Kebodohan.'*

Imam Syafi'i

CONTRIBUTORS

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia

CONTENTS IN BRIEF

1 Teori	1
2 <i>Teori</i>	3
3 Pembangunan Aplikasi	65
4 Pembangunan Aplikasi	95

DAFTAR ISI

Daftar Gambar	xiii
Daftar Tabel	xvii
Foreword	xxi
Kata Pengantar	xxiii
Acknowledgments	xxv
Acronyms	xxvii
Glossary	xxix
List of Symbols	xxxi
Introduction	xxxiii
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
1 Teori	1
2 Teori	3
2.1 Sejarah Python	3
2.2 Implementasi dan Penggunaan Python di Perusahaan Dunia	4

2.3	Jenis-Jenis Variabel	5
2.4	Input dan Output	5
2.5	Operator Aritmatika dan Konversi Tipe Data	6
2.6	Perulangan	6
2.7	Looping For	7
2.8	While	8
2.9	Perulangan Bertingkat	9
2.10	Break	10
2.11	IF Statement	10
2.12	ELSE statement	12
2.13	Elif	13
2.14	Praktek Pertama Python	13
2.14.1	Modulus	13
2.14.2	Hello NPM	14
2.14.3	Hello NPM (3 Digit Belakang)	14
2.14.4	Hello NPM (Digit ke-3)	15
2.14.5	Variabel Alfabet	15
2.14.6	Penjumlahan NPM	15
2.14.7	Perkalian NPM	16
2.14.8	Print Vertical	16
2.14.9	Digit Genap NPM	16
2.14.10	Digit Ganjil NPM	17
2.14.11	Bilangan Prima NPM	17
2.15	Peringatan Error dan Cara Mengatasinya	18
2.16	Except	19
2.17	Run Script Hello World di Spyder	19
2.18	Pemakaian Variable Explorer	20
2.19	Indentasi	20
2.19.1	Penjelasan Indentasi	20
2.20	Jenis-Jenis Error Indentasi	22
2.20.1	Cara Membaca Error	24
2.21	Cara Menangani Error	24
2.22	Quiz: 1	24
2.23	Pip	25
2.23.1	Instalasi Pip	25
2.24	Command Line Interface/Interpreter	27
2.25	Fungsi	28
2.26	Package	28

2.27	Kelas, Objek, Atribut, dan Method	29
2.28	Pemanggilan Library Kelas	30
2.29	Pemakaian Package Fungsi Apabila File Didalam Folder	30
2.30	Pemakaian Package Kelas Apabila File didalam Folder	31
2.31	Praktek Membuat Method dan Pemanggilan Method	31
2.32	Modulus	31
2.33	Hello NPM	31
2.34	Hello NPM (3 Digit Belakang)	32
2.35	Hello NPM (Digit ke-3)	32
2.36	Variabel Alfabet	32
2.37	Penjumlahan NPM	32
2.38	Perkalian NPM	33
2.38.1	Print Vertical	33
2.39	Digit Genap NPM	33
2.40	Digit Ganjil NPM	34
2.41	Bilangan Prima NPM	34
2.42	Pemanggilan Fungsi pada Main.py	34
2.43	Pengelolaan File CSV, Sejarah, dan Contoh	35
2.44	Aplikasi yang bisa Menciptakan File CSV	36
2.45	Menulis dan Membaca File CSV pada Ms.Excel	36
2.46	Sejarah Library CSV	37
2.47	Sejarah Library Pandas	38
2.48	Fungsi - fungsi yang terdapat di library CSV	38
2.49	Fungsi - fungsi yang terdapat di library Pandas	39
2.50	Praktek CSV	39
2.50.1	CSV List	39
2.50.2	CSV Dictionary	40
2.50.3	Hello NPM (Pandas List)	40
2.50.4	Hello NPM (Pandas Dictionary)	40
2.50.5	Pandas Date	40
2.50.6	Pandas Ubah Index	40
2.50.7	Pandas Ubah Column	41
2.50.8	Main CSV	41
2.50.9	Main Pandas	41
2.51	<i>Speech to text</i>	41
2.52	Sejarah speech	42
2.52.1	Perkembangan alat pengenal ucapan	42
2.52.2	Jenis-jenis pengenalan ucapan	43

2.53	Proses kerja alat pengenal ucapan	43
2.54	Aplikasi alat pengenal ucapan	46
2.54.1	Bidang komunikasi	46
2.54.2	Bidang kesehatan	46
2.54.3	Bidang militer	47
2.55	Kelebihan alat pengenal ucapan	47
2.56	Kekurangan alat pengenal ucapan	47
2.57	speech recognition	48
2.57.1	Paket Pengenalan Python Speech	49
2.58	command line interface	49
2.59	Sejarah <i>Selenium</i>	50
2.60	Jenis-Jenis <i>Selenium</i>	51
2.61	Anaconda	53
2.61.1	Instalasi Anaconda	53
2.62	Instalasi Pip	59
2.63	Setting Environment	59
2.63.1	Windows (Windows 10)	59
2.64	Geckodriver dan Chromedriver	62
2.64.1	Gechodriver untuk Mozilla Firefox	62
3	Pembangunan Aplikasi	65
3.1	Pendahuluan	65
4	Pembangunan Aplikasi	95
4.1	Cara penggunaan	96
	Daftar Pustaka	115

DAFTAR GAMBAR

2.1	<i>Launch Spider</i>	20
2.2	<i>Print Hello World</i>	21
2.3	<i>Hello World</i>	22
2.4	<i>Variable Explorer</i>	22
2.5	<i>Indentasi</i>	22
2.6	<i>Error Indentasi</i>	23
2.7	<i>Syntax Error</i>	24
2.8	<i>Syntax yang Telah Diperbaiki</i>	24
2.9	<i>Install pip</i>	26
2.10	<i>Install pip Selesai</i>	26
2.11	<i>Melihat Versi pip</i>	27
2.12	<i>CLI in Command Prompt</i>	28
2.13	Contoh Penulisan CSV pada Excel	37

2.14	Run Setup Anaconda	54
2.15	Setup Loading	54
2.16	Welcome to Anaconda Setup	55
2.17	<i>License Agreement</i>	55
2.18	<i>Just Me(recomended)</i>	56
2.19	<i>Pilih lokasi</i>	56
2.20	<i>Centang Anaconda to my PATH</i>	57
2.21	<i>Installation Complete</i>	57
2.22	<i>Installation Complete</i>	58
2.23	<i>Anaconda+JetBrains</i>	58
2.24	<i>Thanks for install Anaconda</i>	59
2.25	<i>Install pip</i>	59
2.26	<i>Install pip Selesai</i>	59
2.27	<i>Melihat Versi pip</i>	59
2.28	<i>Properties</i>	60
2.29	<i>Advanced system settings</i>	60
2.30	<i>Environment Variables</i>	61
2.31	<i>Path</i>	62
2.32	<i>Gechodriver</i>	63
4.1		96
4.2		97
4.3		97
4.4		98
4.5		99
4.6		100
4.7		101
4.8		102
4.9		103

4.10	104
4.11	105
4.12	106
4.13	107
4.14	108
4.15	109
4.16	110
4.17	111
4.18	112
4.19	113

DAFTAR TABEL

Listings

2.1	Input dan Output	6
2.2	While Loop	6
2.3	For Loop	7
2.4	Nested Loop	7
2.5	if Statement	11
2.6	Elif	11
2.7	Else	11
2.8	Nested If	11
2.9	Modulus	14
2.10	Hello NPM	14
2.11	3 Digit Belakang	14
2.12	Digit ke-3	15
2.13	Variabel Alfabet	15
2.14	Penjumlahan NPM	15
2.15	Perkalian NPM	16
2.16	Print Vertical	16
2.17	Digit Genap NPM	17
2.18	Digit Ganjil NPM	17

2.19 Bilangan Prima NPM	17
2.20 Try Except	19
2.21 Kelas	29
2.22 Modulus	31
2.23 Hello NPM	32
2.24 3 Digit Belakang	32
2.25 Digit ke-3	32
2.26 Variabel Alfabet	32
2.27 Penjumlahan NPM	33
2.28 Perkalian NPM	33
2.29 Print Vertical	33
2.30 Digit Genap NPM	33
2.31 Digit Ganjil NPM	34
2.32 Bilangan Prima NPM	34
2.33 Bilangan Prima NPM	34
2.34 Contoh CSV	35
2.35 Contoh Kode Python to Read CSV	35
2.36 Contoh Kode Python to Read CSV	35
2.37 Contoh CSV	37
src/fcsv.py	38
src/fcsv.py	38
src/fcsv.py	39
src/fcsv.py	39
2.38 CSV List	39
2.39 CSV Dictionary	40
2.40 Pandas List	40
2.41 Pandas Dictionary	40
2.42 Pandas Date	40
2.43 Pandas Ubah Index	40
2.44 Pandas Ubah Column	41
2.45 Main CSV	41
2.46 Digit Genap NPM	41

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

Bandung, Jawa Barat

Februari, 2019

ACKNOWLEDGMENTS

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Intership.

R. M. A.

ACRONYMS

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association

GLOSSARY

git	Merupakan manajemen sumber kode yang dibuat oleh linus torvald.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald

SYMBOLS

A Amplitude

$\&$ Propositional logic symbol

a Filter Coefficient

B Number of Beats

INTRODUCTION

ROLLY MAULANA AWANGGA, S.T., M.T.

Informatics Research Center
Bandung, Jawa Barat, Indonesia

Pada era disruptif saat ini. git merupakan sebuah kebutuhan dalam sebuah organisasi pengembangan perangkat lunak. Buku ini diharapkan bisa menjadi penghantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[1].

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta \sum_{def}^{abc} \quad (I.1)$$

BAB 1

TEORI

BAB 2

TEORI

2.1 Sejarah *Python*

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

Python mendukung multi paradigma pemrograman, utamanya; namun tidak dibatasi; pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembangan

gan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi.

2.2 Implementasi dan Penggunaan Python di Perusahaan Dunia

1. spotify

spotify adalah suatu layanan musik streaming yang menggunakan pemrograman python untuk analisis data dan backend. pada backend spotify berkomunikasi dengan 0MQ. 0MQ itu sendiri adalah suatu framework dan library open source untuk networking. untuk analisis data tersebut spotify menggunakan luigi, dan modul python yang sinkron dengan hadoop.

2. Google

Google ini sudah menggunakan bahasa pemrograman python ini sudah sajak dari awal berdirinya. Dan pada saat ini bahasa pemrograman python merupakan salah satu bahasa pemrograman server-side resmi di google. Meskipun ada script yang ditulis untuk google menggunakan bahasa perl dan bash, maka nantinya script tersebut akan diubah ke python terlebih dahulu, karena kemudahan dalam perawatannya.

3. Industrial Light and Magic

Industrial Light and Magic ini merupakan studio special efek yang dibutuhkan untuk film star wars saja. Karena infrastruktur awal industrial light and magic ini menggunakan C dan C++, maka akan lebih mudah mengintegrasikan bahasa pemrograman python ketimbang bahasa pemrograman lainnya. Dengan menggunakan bahasa pemrograman python ini industrial light and magic dengan mudah membungkus komponen software dan dapat meningkatkan aplikasi grafisnya.

4. Netflix

Netflix adalah suatu layanan pemutaran film yang dapat dilakukan oleh pengguna dimanapun dan kapanpun. Pada netfilx bahasa pemrograman yang digunakan adalah bahasa pemrograman python, bahasa pemrograman ini digunakan pada Central Alert Gateway yang akan me-reroute alert dan mengirimkannya pada individu yang akan melihatnya serta juga dapat secara otomatis reboot atau menghentikan proses yang dianggap bermasalah. Selain itu python juga digunakan untuk menelusuri riwayat dan perubahan pengaturan keamanan.

5. instagram

Instagram adalah suatu aplikasi mobile berbasis IOS, android dan windows phone, dimana pengguna dapat berbagi foto dan video melalui instagram ini. Pada instagram ini menggunakan bahasa pemrograman python dalam task

queuenya atau fitur dimana setiap pengguna dapat berbagi foto atau video ke beberapa social network lainnya seperti facebook, twitter, dan lain-lainnya.

2.3 Jenis-Jenis Variabel

Variabel merupakan tempat penyimpanan data. Tipe data merupakan jenis data yang tersimpan di dalam variabel. terdapat aturan dalam penulisan Variabel.

1. Nama variabel diawali dengan huruf atau garis bawah, contoh: nama, _nama, namaKu, nama_variabel.
2. Karakter selanjutnya dapat berupa huruf, garis bawah atau angka, contoh: __nama, nama1, p1.
3. Karakter bersifat case-sensitive (huruf besar dan huruf kecil dibedakan), contoh: Nama dan NAMA keduanya adalah variabel yang berbeda.
4. Nama variabel tidak boleh menggunakan kata kunci yang ada pada bahasa pemrograman python, contoh: if, else, while
5. Nama variabel tidak boleh diawali dengan angka

Jenis-jenis tipe data pada python. Tipe Data Primitif, dibagi menjadi 3 yaitu:

1. Tipe data integer (angka), penulisannya tidak membutuhkan tanda petik, contoh: 10 atau 15
2. Tipe data string (teks), tipe data string ditandai dengan teks yang diapit oleh tanda petik (""), contoh: "nama saya adalah dinda majesty"
3. Tipe data boolean (memiliki dua nilai yaitu true dan false atau 0 dan 1)

Contoh penulisan variabel dan tipe datanya:

1. angka = 10, angka merupakan nama variabel sedangkan 10 adalah nilai dari variabel yang tipe datanya integer.
2. nama = "Muchamad Innal", nama merupakan nama variabel sedangkan "Muchamad Innal" merupakan nilai dari variabel yang tipe datanya string, ditandai dengan adanya petik ("").
3. makan = True , makan merupakan nama variabel sedangkan True merupakan nilai dari variabel yang tipe datanya boolean.

2.4 Input dan Output

Berikut kode untuk meminta inputan dari user.

```

1 #input output
2 print("masukkan nama anda : ")
3 nama = input()
4 print("nama saya adalah " + nama)

```

Listing 2.1 Input dan Output

Perintah input() berguna untuk meminta inputan dari user, sehingga memungkinkan user untuk menginputkan data.

Perintah print() berguna untuk menampilkan output dari data yang diinputkan oleh user, sehingga data yang diinputkan user dapat ditampilkan ke layar.

2.5 Operator Aritmatika dan Konversi Tipe Data

Operator aritmatika

1. penjumlahan (+)
2. pengurangan (-)
3. perkalian (*)
4. pembagian (/)
5. sisa bagi/modulus (%)
6. pemangkatan (**)

Cara melakukan perubahan terhadap tipe data string menjadi integer, contoh: variabel = "10". Kita dapat mengubah string "10" menjadi angka 10 dengan menambahkan kode int(variabel), dengan begitu 10 yang awalnya bertipe data string akan dikonversikan menjadi integer.

Cara melakukan perubahan terhadap tipe data integer menjadi string, contoh: variabel = 150. Kita dapat mengubah integer 150 menjadi string "150" dengan menambahkan kode str(variabel), maka tipe data dari variabel akan dikonversikan menjadi string.

2.6 Perulangan

perulangan terdiri atas 3 kondisi.

1. While, apabila kondisinya True, maka perulangan akan terus berjalan hingga diperoleh kondisi False. Contoh penggunaan while:

```

1 #perulangan while
2 hitung = 0
3 while (hitung < 9):
4     print ('hitungan ke :', hitung)

```

```

5 hitung = hitung + 1
6
7 print ("Good bye!")

```

Listing 2.2 While Loop

2. For, perulangan for bisa melakukan perulangan terhadap item apapun seperti list atau string. Contoh penggunaan For:

```

1 #perulangan for
2 minum = ["kopi", "susu", "teh"]
3 for minuman in minum:
4     print("Saya suka minum", minuman)

```

Listing 2.3 For Loop

3. nested, perulangan ini memungkinkan adanya perulangan didalam perulangan. Contoh penggunaan nested:

```

1 #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
11 print ("Good bye!")

```

Listing 2.4 Nested Loop

Pada penulisan sintaks While dan For harus memperhatikan identasi (baris yang menjorok ke dalam), jika tidak diperhatikan dengan baik maka akan terjadi error terhadap identasi. Untuk menambahkan identasi dapat menggunakan spasi atau tab.

2.7 Looping For

Seperti di bahasa pemrograman lainnya, Python juga memiliki fungsi for. Bedanya di Python, For tidak hanya untuk perulangan dengan jumlah finite (terbatas), melainkan lebih ke fungsi yang dapat melakukan perulangan pada setiap jenis variabel berupa kumpulan atau urutan. Variabel yang dimaksud bisa berupa list, string, ataupun range. Jika sebuah list atau urutan berisi expression, maka ia akan dievaluasi terlebih dahulu. Kemudian item pertama pada urutan atau list akan diassign sebagai variabel iterating_var. Setelahnya, blok statement akan dieksekusi, berlanjut ke item berikutnya, berulang, hingga seluruh urutan habis. contoh for:

```
for letter in 'Python': # First Example
```

```

print('Current Letter: {}'.format(letter))
fruits = ['banana', 'apple', 'mango']
for fruit in fruits: # Second Example
    print('Current fruit: {}'.format(fruit))

```

Output:

```

Current Letter : P
Current Letter : y
Current Letter : t
Current Letter : h
Current Letter : o
Current Letter : n
Current fruit : banana
Current fruit : apple
Current fruit : mango

```

Anda juga dapat melakukan perulangan berdasarkan indeks atau range dengan memanfaatkan fungsi len():

```

fruits = ['banana', 'apple', 'mango']
for index in range(len(fruits)):
    print('Current fruit: {}'.format(fruits[index]))

```

Output:

```

Current fruit : banana
Current fruit : apple
Current fruit : mango

```

2.8 While

While pada bahasa Python digunakan untuk mengeksekusi statement selama kondisi yang diberikan terpenuhi (True). Kondisi dapat berupa expression apapun, dan harap diingat bahwa True di Python termasuk semua nilai non-zero. Saat kondisi menjadi False, program akan melanjutkan ke baris setelah blok statement.

Seperti for dan semua statement percabangan, blok statement yang mengikuti kondisi while dan memiliki posisi indentasi yang sama, dianggap blok statement yang akan dieksekusi.

Contoh:

```

count = 0
while (count < 5):
    print('The count is: {}'.format(count))
    count = count + 1

```

Output:

```
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
```

Seperti pada bahasa lainnya, eksekusi statement while mungkin bersifat infinit / infinite loop saat sebuah kondisi tidak pernah bernilai False. Contohnya sebagai berikut:

```
var = 1
while var == 1: # This constructs an infinite loop
    num = input('Enter a number: ')
    print('You entered: {}'.format(num))

while True: # This constructs an infinite loop
    num = input('Enter a number: ')
    print('You entered: {}'.format(num))
```

Potongan kode di atas tidak akan pernah bernilai False karena nilai var tidak pernah berubah. Untuk menghentikan infinite loop, gunakan CTRL (atau CMD) - C untuk menghentikannya dan keluar dari program.

Anda juga dapat menyingkat penulisan blok statement While jika statement Anda cukup terwakili oleh satu baris.

```
while (var1): do_something()
```

2.9 Perulangan Bertingkat

Ada kalanya Anda perlu untuk melakukan perulangan bertingkat, misalnya untuk menghasilkan contoh print-out berikut:

```
*****
 ****
 ***
 **
 *
```

Contoh:

```
for i in range(0, 5):
    for j in range(0, 5 - i):
        print('*', end='')
    print()
```

2.10 Break

Pernyataan break menghentikan perulangan kemudian keluar, dilanjutkan dengan mengeksekusi pernyataan (statement) setelah blok perulangan. Salah satu penggunaannya yang paling sering adalah sebuah kondisi eksternal yang membutuhkan program untuk keluar dari perulangan. Jika Anda memiliki perulangan bertingkat, break akan menghentikan perulangan sesuai dengan tingkatan atau di perulangan mana ia berada. Namun jika ia diletakkan di perulangan dengan kedalaman kedua misalnya, hanya perulangan itu saja yang berhenti, tidak dengan perulangan utama. Contoh 1:

```
for letter in 'Python':
    if letter == 'h':
        break
    print('Current letter: {}'.format(letter))
```

Output contoh 1:

```
Current Letter : P
Current Letter : y
Current Letter : t
```

Contoh 2:

```
var = 10
while var > 0:
    print('Current variable value: {}'.format(var))
    var = var - 1
    if var == 5:
        break
```

Output contoh 2:

```
Current variable value : 10
Current variable value : 9
Current variable value : 8
Current variable value : 7
Current variable value : 6
```

2.11 IF Statement

kondisi if dapat digunakan didalam looping dan dapat digunakan untuk memberikan kondisi tertentu dengan cara mengetikkan if lalu kondisi yang akan terjadi.

1. if hanya menjalankan satu kondisi dan menampilkan satu output. Contoh: kondisi dimana variabel a lebih besar dari variabel b, maka tampilkan hasil bahwa a lebih besar dari b.

```

1 #if statement
2 a = 330
3 b = 200
4 if a > a:
5     print("a lebih besar dari b")

```

Listing 2.5 if Statement

2. elif digunakan apabila kondisi pertama tidak benar maka lakukan kondisi lain (alternatif). Contoh: kondisi dimana variabel a sama dengan variabel b, maka jika b lebih besar dari a, tampilkan hasil b lebih besar dari a, namun jika a dan b bernilai sama, maka tampilkan a sama dengan b

```

1 #elif
2 a = 33
3 b = 33
4 if b > a:
5     print("b lebih besar dari a")
6 elif a == b:
7     print("a sama dengan b")

```

Listing 2.6 Elif

3. else digunakan apabila kondisi yang terjadi bernilai salah, maka lakukan else. Contoh: kondisi dimana variabel a lebih besar dari variabel b, maka jika b lebih besar dari a, tampilkan hasil b lebih besar dari a, jika a dan b bernilai sama, maka tampilkan a sama dengan b, jika salah maka tampilkan a lebih besar dari pada b

```

1 #else
2 a = 200
3 b = 33
4 if b > a:
5     print("b is greater than a")
6 elif a == b:
7     print("a and b are equal")
8 else:
9     print("a is greater than b")

```

Listing 2.7 Else

4. Nested if merupakan if didalam if (if bersarang), terdapat dua if didalam satu kondisi. Contoh: variabel x sama dengan 41, kondisi pertama yaitu jika x besar dari 10 maka tampilkan lebih besar dari 10, kondisi kedua yaitu jika x besar dari 20, maka tampilkan lebih besar dari 20, jika salah maka tampilkan tidak melebihi 20.

```

1 #nested if
2 x = 41
3
4 if x > 10:
5     print("lebih besar dari 10,")
6     if x > 20:

```

```

7     print("lebih besar dari 20!")
8 else:
9     print("tidak melebihi 20.")

```

Listing 2.8 Nested If

contoh if statement:

```

var1 = 100
if var1:
    print ("1 - Got a true expression value")
    print (var1)
var2 = 0
if var2:
    print ("2 - Got a true expression value")
    print (var2)

```

Output:

```

1 - Got a true expression value
100

```

2.12 ELSE statement

Statement Else dapat dikombinasikan dengan IF Statement, sebagai jalan keluar saat kondisi / hasil evaluasi bernilai False. Else bersifat opsional dan tunggal. contoh else statement:

```

amount = int(input("Enter amount: "))
if amount<1000:
    discount = amount*0.05
    print ("Discount",discount)
else:
    discount = amount*0.10
    print ("Discount",discount)

print ("Net payable:",amount-discount)

```

Output true:

```

Enter amount: 600
Discount 30.0
Net payable: 570.0

```

Output false:

```

Enter amount: 1200
Discount 120.0
Net payable: 1080.0

```

2.13 Elif

Alternatif untuk Switch/Case dan IF bertingkat di python. Elif adalah kependekan dari else if, dan merupakan alternatif untuk if bertingkat atau switch/case di beberapa bahasa pemrograman lain. Sebuah IF Statement dapat diikuti satu atau lebih statement elif (opsional dan tidak dibatasi). contoh elif:

```
amount = int(input("Enter amount: "))
if amount<1000:
    discount = amount*0.05
    print ("Discount",discount)
elif amount<5000:
    discount = amount*0.10
    print ("Discount",discount)
else:
    discount = amount*0.15
    print ("Discount",discount)
print ("Net payable:",amount-discount)
```

Output true if:

```
Enter amount: 600
Discount 30.0
Net payable: 570.0
```

Output true elif:

```
Enter amount: 3000
Discount 300.0
Net payable: 2700.0
```

Output false:

```
Enter amount: 6000
Discount 900.0
Net payable: 5100.0
```

2.14 Praktek Pertama Python

2.14.1 Modulus

Praktek kali ini kita akan mencoba praktek menggunakan modulus atau sisa bagi, kita membuat inputan terlebih dahulu menggunakan perintah input. Kemudian buatlah variabel untuk menampung hasil sisa bagi dari jumlah yang diinputkan. Misalnya, teman-teman menginputkan nilai yaitu 1184011, maka hasil modulus 1184011 mod 3 adalah 1 maka print 1184011 menggunakan tanda pagar. jika hasil modulus adalah 0 maka print 1184011 menggunakan tanda bintang.

```

1 #Modulus
2 print("Soal no 1")
3
4 print("Masukkan NPM anda: ")
5 NPM = input()
6
7 npm = int(NPM) % 3
8 print(npm)
9
10 print("###    ###    #####    ###    ###    #####    ###    ###")
11 print("###    ###    #####    ###    ###    #####    ###    ###")
12 print("###    ###    ###    ###    ###    ###    ###    ###")
13 print("###    ###    ###    ###    ###    ###    ###    ###")
14 print("###    ###    #####    #####    ###    ###    ###    ###")
15 print("###    ###    ###    ###    ###    ###    ###    ###")
16 print("###    ###    ###    ###    ###    ###    ###    ###")
17 print("###    ###    #####    #####    ###    ###    ###    ###")
18 print("###    ###    #####    #####    ###    ###    ###    ###")

```

Listing 2.9 Modulus

2.14.2 Hello NPM

Setelah selesai praktek modulus kita akan menampilkan output berupa kalimat "Hello 1184011 Apa Kabar" sebanyak 2 digit belakang angka, yaitu angka 11, maka akan berulang sebanyak 11 kali.

```

1 #Hello NPM
2 print("Soal No 2")
3
4 Loop = NPM[5:7]
5
6 for x in range(int(Loop)):
7     print("Hallo " + NPM + " Apa Kabar?")

```

Listing 2.10 Hello NPM

2.14.3 Hello NPM (3 Digit Belakang)

Jika telah selesai, maka selanjutnya kita akan melakukan praktek untuk menampilkan output berupa kalimat "Hallo 011 Apa Kabar?" sebanyak angka keenam ditambah angka ketujuh atau 1 ditambah 1, sehingga kalimat tersebut akan berulang sebanyak 2 kali.

```

1 #3 digit belakang NPM
2 print("Soal No 2")
3
4 Loop = NPM[4:7]
5
6 total = int(NPM[5]) + int(NPM[6])
7
8 for x in range(total):

```

```
9     print("Hallo " + Loop + " Apa Kabar?")
```

Listing 2.11 3 Digit Belakang

2.14.4 Hello NPM (Digit ke-3)

Kemudian kita akan melakukan praktek untuk menampilkan output berupa kalimat "Hallo 0 Apa Kabar?".

```
1 #digit ke 3
2 print("Soal No 3")
3
4 Loop = NPM[4]
5 print("Hello " + Loop + " Apa Kabar?")
```

Listing 2.12 Digit ke-3

2.14.5 Variabel Alfabet

Teman-teman, sekarang mari kita coba menambahkan variabel pada angka 1184011, tambahkan abcdefg kedalam sebuah variabel dan tambahkan variabel index dengan nilai 0, buatlah perulangan agar variabel huruf menyesuaikan dengan variabel angka sehingga menjadi a=1, b=1, c=8, d=4, e=0, f=1 ,g=1.

```
1 #variabel alfabet
2 print("Soal no 5")
3
4 var = "abcdefg"
5 index = 0
6
7 for i in var:
8     print(i + " = " + NPM[index])
9     index += 1
```

Listing 2.13 Variabel Alfabet

2.14.6 Penjumlahan NPM

Praktek selanjutnya adalah menjumlahkan angka 1184011 dengan menerapkan perulangan dan penambahan. Apabila nilai 1 telah didapatkan maka akan ditambahkan dengan nilai 1 sehingga menjadi 2, kemudian nilai 2 ditambahkan lagi dengan nilai selanjutnya yaitu 8 sehingga menjadi 10, begitu seterusnya.

```
1 #penjumlahan NPM
2 print("Soal no 6")
3
4 index = 0
5 angka = 0
6
7 for i in NPM:
8     jumlah = int(NPM[index]) + int(angka)
```

```

9     angka = jumlah
10    index += 1
11
12 print(jumlah)

```

Listing 2.14 Penjumlahan NPM

2.14.7 Perkalian NPM

Praktek selanjutnya adalah mengalikan angka 1184011 dengan menerapkan perulangan dan perkalian. Apabila nilai 1 telah didapatkan maka akan dikalikan dengan nilai 1 sehingga menjadi 1, kemudian nilai 1 dikalikan lagi dengan nilai selanjutnya yaitu 8 sehingga menjadi 8, begitu seterusnya.

```

1 #perkalian_NPM
2 print("Soal no 7")
3
4 index = 0
5 angka = 0
6
7 for i in NPM:
8     jumlah = int(NPM[index]) * int(angka)
9     angka = jumlah
10    index += 1
11
12 print(jumlah)

```

Listing 2.15 Perkalian NPM

2.14.8 Print Vertical

Melakukan print secara vertikal hanya perlu menambahkan perulangan terhadap nilai 1184011.

```

1 #print_vertical
2 print("Soal no 8")
3
4 for i in NPM:
5     print(i)

```

Listing 2.16 Print Vertical

2.14.9 Digit Genap NPM

Selanjutnya, mari kita lakukan print hanya terhadap digit genap pada angka 1184011 dengan memanfaatkan perulangan, if statement dan operator logika. Logika yang akan diterapkan adalah masing-masing angka 1184011 akan di cek terlebih dahulu, apakah angka tersebut memiliki angka yang apabila dibagi 2 akan menghasilkan sisa bagi sama dengan 0 dan angka tersebut tidak boleh sama dengan 0, karena angka 0 bukan merupakan angka ganjil maupun genap.

```

1 #digit genap NPM
2 print("Soal no 9")
3
4 index = 0
5 for i in NPM:
6     if (int(NPM[index])%2 == 0) & (int(NPM[index]) != 0):
7         print(NPM[index])
8     index += 1

```

Listing 2.17 Digit Genap NPM

2.14.10 Digit Ganjil NPM

Selanjutnya, mari kita lakukan print hanya terhadap digit ganjil pada angka 1184011 dengan memanfaatkan perulangan, if statement dan operator logika. Logika yang akan diterapkan adalah masing-masing angka 1184011 akan di cek terlebih dahulu, apakah angka tersebut memiliki angka yang apabila dibagi 2 akan menghasilkan sisa bagi tidak sama dengan 0 dan angka tersebut tidak boleh sama dengan 0, karena angka 0 bukan merupakan angka ganjil maupun genap.

```

1 #digit ganjil NPM
2 print("Soal no 10")
3
4 index = 0
5 for i in NPM:
6     if (int(NPM[index])%2 != 0) & (int(NPM[index]) != 0):
7         print(NPM[index])
8     index += 1

```

Listing 2.18 Digit Ganjil NPM

2.14.11 Bilangan Prima NPM

Praktek terakhir adalah menampilkan hasil dari bilangan prima angka 1184011 dengan cara apabila angka 1184011 memiliki angka yang merupakan angka prima maka akan ditampilkan, logika yang akan diterapkan adalah apabila angka kecil sama dengan angka 1 maka angka tersebut bukan bilangan prima. Jika angka 1184011 dibagi 2 setelah itu dibagi dengan angka itu sendiri dan menghasilkan sisa bagi sama dengan 0, maka angka tersebut bukan bilangan prima

```

1 #bilangan prima NPM
2 print("Soal no 11")
3 index = 0
4
5 for i in NPM:
6     prima = True
7     var=int(NPM[index])
8     if(var<=1):
9         prima=False
10    for i in range (2 ,var):
11        if( var%i==0):
12            prima=False

```

```

13 if(prima==True):
14     print(var,"Prima")
15 else:
16     print(var,"bukan prima")
17 index += 1

```

Listing 2.19 Bilangan Prima NPM

2.15 Peringatan Error dan Cara Mengatasinya

1. NameError, terjadi apabila kode mengeksekusi nama yang tidak terdefenisikan. Contoh:

```

nama = "Muchamad Innal"
print(Nama)

```

Maka akan menghasilkan output NameError: name 'Nama' is not defined. error ini dapat diatasi dengan mengubah variabel yang di print sesuai dengan variabel yang didefinisikan, karena penulisan pada python bersifat case-sensitive

2. SyntaxError, terjadi apabila kode python mengalami kesalahan saat penulisan. Contoh: menuliskan variabel yang didahului angka (1nama = "Muchamad Innal") maka akan muncul eror SyntaxError: invalid syntax. error ini dapat diatasi dengan memperhatikan tata cara penulisan kode pada bahasa pemrograman python.
3. TypeError, terjadi apabila kode melakukan operasi atau fungsi terhadap tipe data yang tidak sesuai. Contoh: melakukan penjumlahan terhadap tipe data string dan integer. eror ini dapat diatasi dengan mengubah tipe data string menjadi integer.

```

a = "10"
b = 5

print(a + b)

```

Maka akan menghasilkan output eror TypeError: can only concatenate str (not "int") to str

4. IndentationError, terjadi apabila kode perulangan atau pengkondisian tidak menjorok kedalam (tidak menggunakan identasi), error ini dapat diatasi dengan menambahkan tab atau spasi. Contoh:

```
a = 200
```

```
b = 330
if b > a:
print("b lebih besar dari a")
```

Maka akan menghasilkan output eror IndentationError: expected an indented block

2.16 Except

Try Except merupakan perintah yang bisa digunakan dalam penanganan error pada bahasa pemrograman python. perintah ini biasanya digunakan saat penanganan error input/output, operasi database, pengaksesan indeks suatu list atau dictionary dan berbagai kasus lainnya.

Contoh sederhana penggunaan Try-Except saat menangani NameError

```
1 #try except
2 try:
3     print(x)
4 except NameError:
5     print("Variable x tidak ada")
6 except:
7     print("ada sesuatu yang salah nih")
```

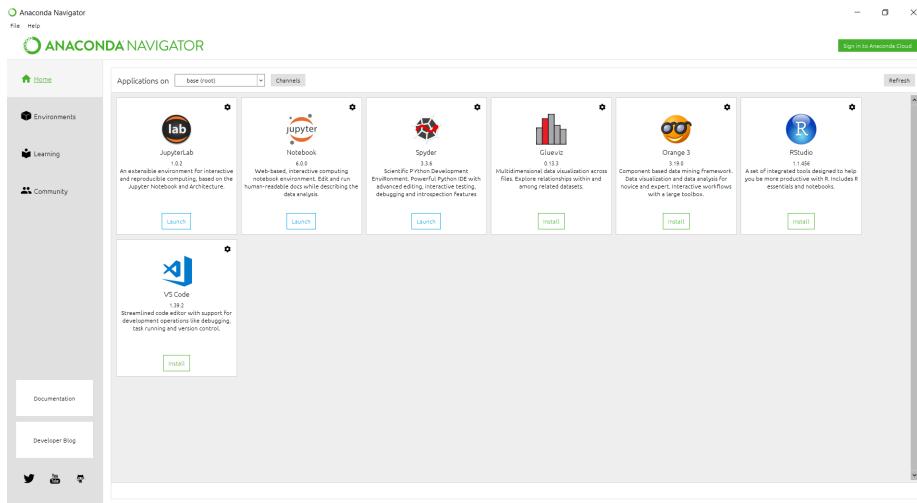
Listing 2.20 Try Except

Pada contoh diatas, try except akan menghasilkan output Variabel x tidak ada, karena kita tidak mendefinisikan variabel x sebelum kita menampilkan output x ke layar. Dengan menggunakan try except kode yang kita buat terhindar dari error dan kita bisa mengetahui kesalahan yang terjadi ketika terdapat error pada kode kita.

2.17 Run Script Hello World di Spyder

Jika teman-teman telah berhasil melakukan instalasi anaconda3 maka teman-teman akan memiliki software spyder, pada software ini teman-teman akan mengetikkan kode program yang akan teman-teman buat nantinya. Sekarang ayo kita coba menjalankan program untuk mencetak kata *hello world*.

1. Buka anaconda navigator, lalu klik launch seperti pada gambar 2.1.
2. ketikkan `print("Hello World")` dan run spyder dengan cara mengklik tombol berwarna hijau yang terletak ditengah toolbar, untuk lebih jelas dapat dilihat pada gambar 2.2.
3. hasilnya akan tampak seperti pada gambar 2.3



Gambar 2.1 *Launch Spider*

2.18 Pemakaian Variable Explorer

Variable explorer akan secara otomatis terisi ketika kita membuat variable, pada variable explorer kita bisa melihat nama variable, tipe data, length, dan value dari variable tersebut. Contoh penggunaan variabel explorer dapat teman-teman lihat pada gambar 2.4.

2.19 Indentasi

2.19.1 Penjelasan Indentasi

Indentasi adalah bagian paragraf yang menjorok ke dalam pada baris-baris paragraf. Mengatur indentasi dapat menggunakan tab atau spasi. Indentasi digunakan oleh bahasa pemrograman python sebagai pengganti briket () untuk membuka dan menutup fungsi. Error indentasi dapat terjadi apabila syntax tidak menggunakan tab atau space. Contoh yang benar (menggunakan tab/spasi sebagai indentasi):

```
# blok percabangan if
if username == 'petanikode':
    print("Selamat Datang Admin")
    print("Silahkan ambil tempat duduk")

# blok percabangan for
for i in range(10):
    print i
```

Activities Spyder ▾

File Edit Search Source Run Debug Consoles Projects To

Editor - /home/burger-man/.config/spyder-py3/temp.py

temp.py ✘

```
1 # -*- coding: utf-8 -*-
2 """
3 Spyder Editor
4
5 This is a temporary script file.
6 """
7
8 hello = "Hello World"
9
10 print(hello)
```

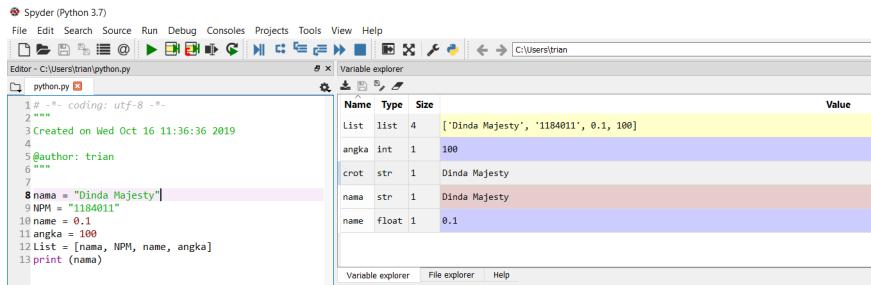
Run file

```
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
```

IPython 7.6.1 -- An enhanced Interactive Python.

```
In [1]: runfile('C:/Users/trian/.spyder-py3/temp.py', wdir='C:/Users/trian/.spyder-py3')
Hello World
```

Gambar 2.3 Hello World



Gambar 2.4 Variable Explorer

Contoh yang salah (tidak menggunakan tab/spasi):

```
# blok percabangan if
if username == 'petanikode':
print("Selamat Datang Admin")
print("Silahkan ambil tempat duduk")

# blok percabangan for
for i in range(10):
print i
```

2.20 Jenis-Jenis Error Indentasi

IndentationError: unexpected indent. Error diatas terjadi apabila syntax kekurangan tab atau spasi. Contoh error identasi dapat dilihat pada gambar 2.5. Apabila di



Gambar 2.5 Indentasi

running akan memunculkan error seperti pada gambar 2.6.

The screenshot shows the Spyder Python IDE interface. On the left is a vertical docked pane titled "Activities" containing icons for various applications: a folder, TeX Maker, a magnifying glass, a browser, a terminal, and a spider web. The main window title is "Spyder". The menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. Below the menu is a toolbar with icons for file operations like Open, Save, and Print, along with run and debug buttons. The central area is the "Editor - /home/burger-man/.config/spyder-py3/temp.py" window, which displays the following code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Spyder Editor
4
5 This is a temporary script file.
6 """
7
8 hello = "Hello World"
9
10 if hello == "Hello World":
11     print (hello)
12 else :
13     print (hello)
```

The line "11 print (hello)" is highlighted with a yellow triangle icon, indicating an error in the indentation of the print statement.

Gambar 2.6 Error Indentasi

2.20.1 Cara Membaca Error

Jika terjadi error maka cari di line berapa error terjadi, pada gambar 2.6 terdapat error indentasi pada line 10.

2.21 Cara Menangani Error

Menangani error indentasi dapat dilakukan dengan cara menambahkan tab atau space pada line yang error. Untuk lebih jelasnya dapat teman-teman lihat pada gambar 2.7.

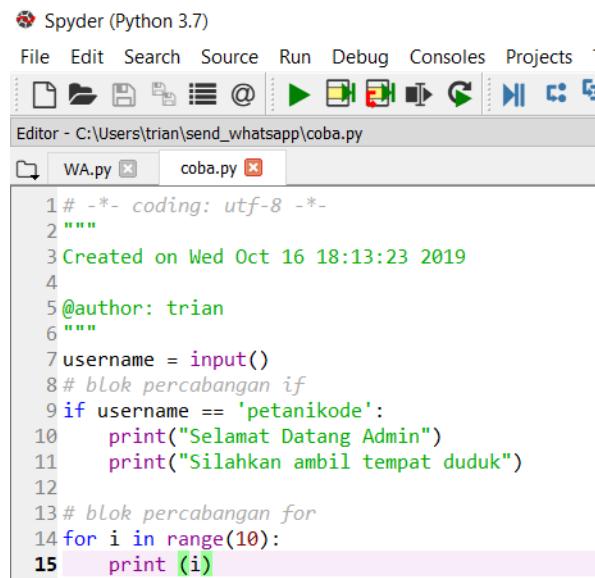
```

1
2
3
4
5 if statut == 1:
6     print("berhasil")
7     print("Data berhasil diperbarui")
8
9     print("Gagal!")
10    print("Data gagal diperbarui")

```

Gambar 2.7 Syntax Error

Penulisan syntax identasi yang benar dapat dilihat pada gambar 2.8.



```

Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects 1
Editor - C:\Users\trian\send_whatsapp\coba.py
WA.py coba.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Oct 16 18:13:23 2019
4
5 @author: trian
6 """
7 username = input()
8 # blok percabangan if
9 if username == 'petanikode':
10     print("Selamat Datang Admin")
11     print("Silahkan ambil tempat duduk")
12
13 # blok percabangan for
14 for i in range(10):
15     print (i)

```

Gambar 2.8 Syntax yang Telah Diperbaiki

2.22 Quiz: 1

- Siapakah pencipta Python?
 - Guido van Rossum

- b) Guido van Persie
 - c) Guido van Linguini
 - d) Guido van Laptop
2. Pada tahun berapa python dikembangkan?
- a) 1999
 - b) 1995
 - c) 1945
 - d) 1990
3. Apa itu indentasi?
- a) Bagian paragraf yang menjorok ke dalam
 - b) Bagian paragraf yang menjorok ke sungai
 - c) Bagian paragraf yang menjorok ke laut
 - d) Bagian paragraf yang menjorok ke hati
4. Apa itu try except?
- a) Perintah untuk penanganan masalah hidup
 - b) Perintah untuk penanganan masalah error
 - c) Perintah untuk penanganan masalah galau
 - d) Perintah untuk penanganan masalah sakit perut
5. Apa itu if statement?
- a) Kondisi yang digunakan untuk percabangan logika
 - b) Kondisi yang digunakan untuk percabangan pohon
 - c) Kondisi yang digunakan untuk percabangan tunas
 - d) Kondisi yang digunakan untuk percabangan akar

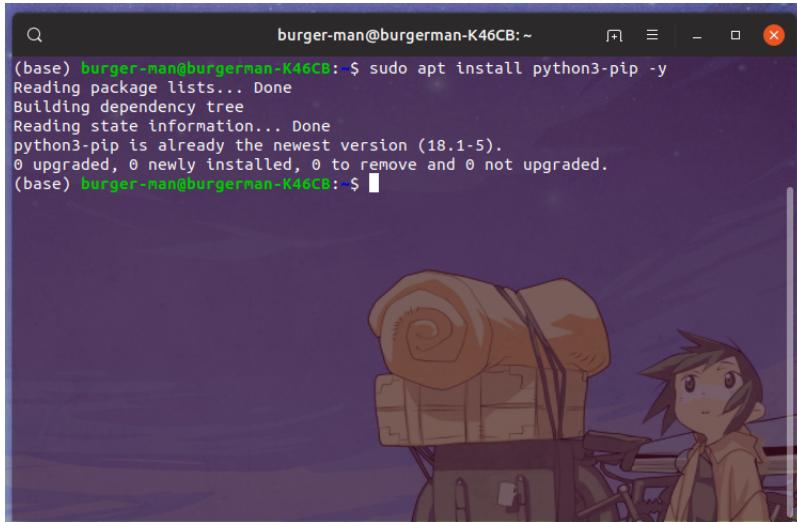
2.23 Pip

Pip merupakan modul atau paket yang harus kita miliki apabila kita menggunakan bahasa pemrograman python. Pip digunakan untuk menginstal package-package python yang akan kita gunakan dalam pembuatan kode program.

2.23.1 Instalasi Pip

Cara melakukan instalasi pip pada anaconda CLI:

1. buka anaconda prompt (Start -> Anaconda Prompt)
2. ketikkan conda install -c anaconda pip seperti pada gambar 2.9.



Gambar 2.9 *Install pip*

3. ketik y, lalu enter. Tunggu hingga proses instalasi selesai seperti pada gambar 2.10.

```

Anaconda Prompt (Anaconda3)
Total: 10.9 MB

The following packages will be UPDATED:
  conda           pkgs/main::conda-4.7.10-py37_0 --> anaconda::conda-4.7.12-py37_0

The following packages will be SUPERSEDED by a higher-priority channel:
  ca-certificates      pkgs/main --> anaconda
  certifi             pkgs/main --> anaconda
  openssl             pkgs/main --> anaconda
  pip                 pkgs/main --> anaconda

?proceed {[y]/n}? y

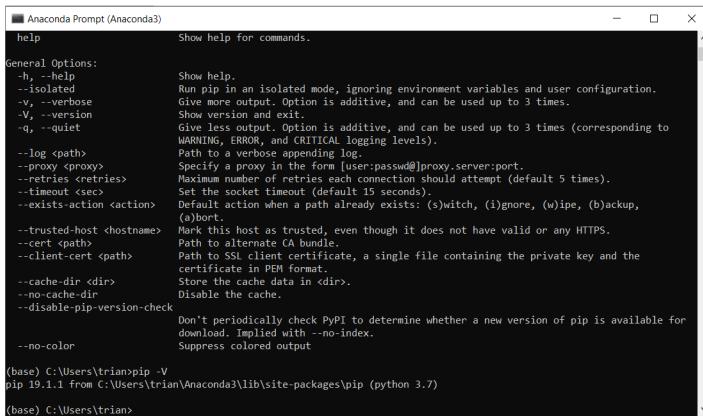
Downloading and Extracting Packages
openssl-1.1.1c          | 5.7 MB  | #####| 100%
certifi-2019.6.16        | 155 KB  | #####| 100%
ca-certificates-2019     | 166 KB  | #####| 100%
pip-19.1.1               | 1.8 MB  | #####| 100%
conda-4.7.12             | 3.0 MB  | #####| 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(base) C:\Users\trian>

```

Gambar 2.10 *Install pip Selesai*

4. jika telah selesai, lakukan pengecekan versi pip dengan mengetikkan pip -V seperti pada gambar 2.11.



```

Anaconda Prompt (Anaconda3)
help           Show help for commands.

General Options:
-h, --help      Show help.
--isolated     Run pip in an isolated mode, ignoring environment variables and user configuration.
-v, --verbose   Give more output. Option is additive, and can be used up to 3 times.
-V, --version   Shows version and exit.
-q, --quiet     Give less output. Option is additive, and can be used up to 3 times (corresponding to
                WARNING, ERROR, and CRITICAL logging levels).
--log <path>   Path to a verbose appending log.
--proxy <proxy> Specify a proxy in the form [user:password@]proxy.server:port.
--retries <retries> Maximum number of retries each connection should attempt (default 5 times).
--timeout <sec> Set the socket timeout (default 15 seconds).
--exists-action <action> Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup,
                    (a)bort.
--trusted-host <hostname> Mark this host as trusted, even though it does not have valid or any HTTPS.
--cert <path>   Path to alternative CA bundle.
--client-cert <path> Path to an SSL client certificate, a single file containing the private key and the
                  certificate in PEM format.
--cache-dir <dir> Store the cache data in <dir>.
--no-cache-dir Disable the cache.
--disable-pip-version-check Don't periodically check PyPI to determine whether a new version of pip is available for
                                download. Implied with --no-index.
--no-color      Suppress colored output

(base) C:\Users\trian>pip -V
pip 19.1.1 from C:\Users\trian\Anaconda3\lib\site-packages\pip (python 3.7)

(base) C:\Users\trian>

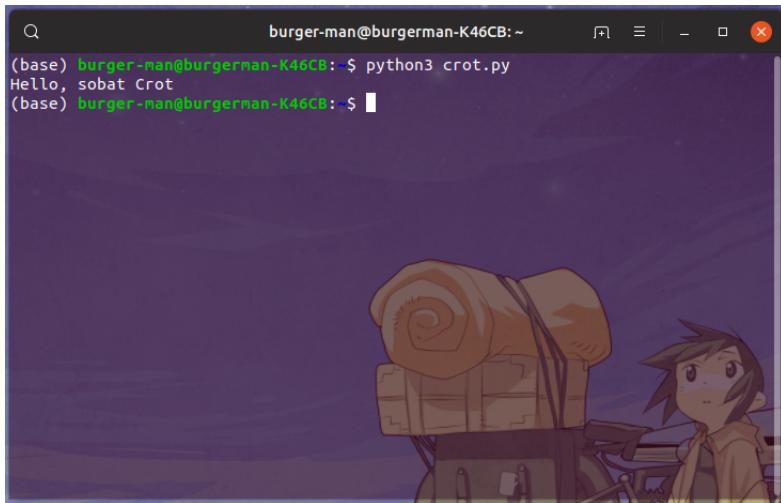
```

Gambar 2.11 Melihat Versi pip

2.24 Command Line Interface/Interpreter

Command line interpreter atau yang biasa teman-teman ketahui sebagai command prompt memungkinkan kita untuk menuliskan baris perintah yang akan dijalankan oleh komputer. Command line interpreter hanya berupa script atau tulisan kode program, berbeda dengan graphich user interface atau GUI yang memungkinkan kita memerintahkan sesuatu kepada komputer dengan hanya menggunakan tombol-tombol dan tampilan yang mudah dipahami. Cara menjalankan program python pada CLI:

1. Buka command prompt lalu ketikkan python seperti pada gambar 2.12.
2. Buatlah perintah print, input, perkalian, dan pembagian seperti pada gambar 2.12.
3. Bisa juga menjalankan file .py yang telah dibuat di IDE dengan cara python namafile.py, lalu klik enter seperti pada gambar 2.12.



Gambar 2.12 *CLI in Command Prompt*

2.25 Fungsi

Fungsi adalah sebuah blok kode yang memiliki nama fungsi dan kode program di dalamnya. Fungsi dapat dipanggil berkali-kali sesuai dengan nama fungsi yang telah didefinisikan. Fungsi memiliki nilai kembalian (return).

Contoh fungsi:

```
def my biodata(nama, umur) :  
    bio = "nama saya " + nama + " umur saya " + umur  
    return bio
```

Inputan pada fungsi berada di dalam (). Contoh (str), ini merupakan inputan yang terdapat pada fungsi. Return merupakan kembalian dari fungsi. Misalnya return nama, maka program akan mengembalikan string yang terdapat dalam variabel nama yaitu "Nama Saya Dinda Majesty".

2.26 Package

Package merupakan sekumpulan modul yang dikemas oleh programmer dengan tujuan agar mempermudah dalam pembuatan kode program. Kita dapat membuat sebuah kode program atau fungsi didalamnya dan dapat secara mudah menggunakan kode program itu dengan cara memanggilnya pada kode program lainnya atau import package.

Contoh package:

```
def my biodata(nama, umur) :
```

```

        bio = "nama saya " + nama + " umur saya " + umur
        return bio
    def my_study(kampus, prodi):
        study = "saya berkuliahan di " + kampus + " program studi " + prodi
        return study

```

Kode diatas merupakan isi dari file fungsi.py, sedangkan saya ingin menjalankan program fungsi.py pada main.py sehingga kode program pada file main.py akan dituliskan seperti berikut:

```

import fungsi

nama = "Muchamad Innal"
umur = "19 Tahun"
biodata = my_biodata(nama, umur)
print(biodata)

kampus = "Politeknik Pos Indonesia"
prodi = "D4-Teknik Informatika"
kuliah = my_study(kampus, prodi)
print(kuliah)

```

Kode program pada file main.py akan mengimport kode program yang ada pada file fungsi.py, sehingga dengan adanya fungsi dan package kita dapat dengan mudah melakukan pemanggilan fungsi yang telah kita deskripsikan sebelumnya, walaupun berada pada file python yang berbeda.

2.27 Kelas, Objek, Atribut, dan Method

Kelas merupakan blueprint dari sebuah objek atau kode program yang berisi fungsi dan dibuat untuk mendefenisikan objek dengan atribut yang sesuai dengan kelas yang telah dibuat.

Objek merupakan wujud dari kelas. Sebuah kelas harus memiliki objek yang nantinya akan dikodekan sesuai dengan fungsi yang telah dibuat pada kelas, tanpa adanya objek sebuah kelas tidak akan bisa menjalankan fungsi-fungsi didalamnya.

Atribut berisi variabel yang memiliki tipe data dan dapat kita berikan pada objek.

Method merupakan kode program yang berisi tindakan atau perintah untuk menjalankan objek.

contoh pembuatan kelas, objek, atribut, dan method:

```

1 class Kelas:
2     def __init__(self, NPM):
3         self.NPM = NPM
4
5     def Modulus(self):
6         #Modulus
7         print("Soal no 1")

```

```

8      npm = int(self.NPM) % 3
9      print(npm)
10
11      print("###    ###    #####    ###    ###    #####    ###")
12      print("###    ###    #####    ###    ###    #####    ###")
13      print("###    ###    ###    ###    ###    ###    ###    ###")
14      print("###    ###    ###    ###    ###    ###    ###    ###")
15      print("###    ###    ###    ###    ###    ###    ###    ###")
16      print("###    ###    #####    #####    ###    ###    ###")
17      print("###    ###    ###    ###    ###    ###    ###    ###")
18      print("###    ###    ###    ###    ###    ###    ###    ###")
19      print("###    ###    #####    ###    ###    #####    ###")
20      print("###    ###    #####    ###    ###    #####    ###")

```

Listing 2.21 Kelas

2.28 Pemanggilan Library Kelas

Pemanggilan library kelas dapat dilakukan dengan cara import dan membuat objek dari kelas tersebut.

Contohnya, kita memiliki file python yang diberi nama ngitung dan didalamnya terdapat class Ngitung yang memiliki banyak fungsi didalamnya. Untuk melakukan pemanggilan class maka kita bisa mengetikkan kode seperti berikut.

```

import ngitung

hitung = ngitung.Ngitung

```

2.29 Pemakaian Package Fungsi Apabila File Didalam Folder

Pemakaian Package fungsi apabila file terdapat didalam sebuah folder maka kita bisa menggunakan from folder import file dan from file import fungsi. Contohnya, kita memiliki folder src yang didalamnya terdapat file fungsi.py dan didalam fungsi.py terdapat fungsi Berhitung, untuk mengimportkan fungsi maka kita dapat mengetikkan kode seperti berikut.

```

from src import fungsi
from fungsi import Berhitung

```

2.30 Pemakaian Package Kelas Apabila File didalam Folder

Pemakaian package kelas apabila file terdapat didalam sebuah folder maka kita bisa menggunakan from folder import file dan from file import kelas. Contohnya, kita memiliki folder src yang didalamnya terdapat file fungsi.py dan didalam fungsi.py terdapat kelas Ngitung, maka untuk melakukan import kelas kita dapat mengetikkan kode sebagai berikut.

```
from src import fungsi
Kelas = fungsi.Ngitung(a,b)
```

2.31 Pretek Membuat Method dan Pemanggilan Method

Cara membuat method hanya dengan menambahkan def Nama Method, self atau variabel inputan, dan indentation pada source code yang telah dipraktekkan sebelumnya. Pembuatan method merupakan hal yang wajib saat kita membuat sebuah program yang berorientasi objek (OOP).

2.32 Modulus

```
1  def Modulus( self ) :
2      #Modulus
3      print("Soal no 1")
4
5      npm = int( self.NPM ) % 3
6      print(npm)
7
8      print("###    ###    #####    ###    ###    #####    ###"
9           "###")
10     print("###    ###    #####    ###    ###    #####    ###"
11          "###")
12     print("###    ###    ###    ###    ###    ###    ###    ###"
13          "###")
14     print("###    ###    #####    #####    ###    ###    ###"
15          "###")
16     print("###    ###    ###    ###    ###    ###    ###    ###"
17          "###")
```

Listing 2.22 Modulus

2.33 Hello NPM

```

1 def Hello_NPM(self):
2     #Hello NPM
3     print("Soal No 2")
4
5     Loop = self.NPM[5:7]
6
7     for x in range(int(Loop)):
8         print("Hallo " + self.NPM + " Apa Kabar?")

```

Listing 2.23 Hello NPM

2.34 Hello NPM (3 Digit Belakang)

```

1 def NPM_DigitBelakang(self):
2     #3 digit belakang NPM
3     print("Soal No 3")
4
5     Loop = self.NPM[4:7]
6
7     total = int(self.NPM[5]) + int(self.NPM[6])
8
9     for x in range(total):
10        print("Hallo " + Loop + " Apa Kabar?")

```

Listing 2.24 3 Digit Belakang

2.35 Hello NPM (Digit ke-3)

```

1 def NPM_DigitKetiga(self):
2     #digit ke 3
3     print("Soal No 4")
4
5     Loop = self.NPM[4]
6     print("Hello " + Loop + " Apa Kabar?")

```

Listing 2.25 Digit ke-3

2.36 Variabel Alfabet

```

1 def Variabel_Alfabet(self):
2     #variabel alfabet
3     print("Soal no 5")
4
5     var = "abcdefg"
6     index = 0
7
8     for i in var:
9         print(i + " = " + self.NPM[index])
10        index += 1

```

Listing 2.26 Variabel Alfabet

2.37 Penjumlahan NPM

```

1 def Penjumlahan_NPM(self):
2     #penjumlahan NPM
3     print("Soal no 6")
4
5     index = 0
6     angka = 0
7
8     for i in self.NPM:
9         jumlah = int(self.NPM[index]) + int(angka)
10        angka = jumlah
11        index += 1
12
13    print(jumlah)

```

Listing 2.27 Penjumlahan NPM

2.38 Perkalian NPM

```

1 def Perkalian_NPM(self):
2     #perkalian NPM
3     print("Soal no 7")
4
5     index = 0
6     angka = 0
7
8     for i in self.NPM:
9         jumlah = int(self.NPM[index]) * int(angka)
10        angka = jumlah
11        index += 1
12
13    print(jumlah)

```

Listing 2.28 Perkalian NPM

2.38.1 Print Vertical

```

1 def Print_Vertical(self):
2     #print vertical
3     print("Soal no 8")
4
5     for i in self.NPM:
6         print(i)

```

Listing 2.29 Print Vertical

2.39 Digit Genap NPM

```

1 def DigitGenap(self):
2     #digit genap NPM
3     print("Soal no 9")
4
5     index = 0
6     for i in self.NPM:
7         if (int(self.NPM[index])%2 == 0) & (int(self.NPM[index])
8             != 0):

```

```

8     print(self.NPM[index])
9     index += 1

```

Listing 2.30 Digit Genap NPM

2.40 Digit Ganjil NPM

```

1 def DigitGanjil(self):
2     #digit ganjil NPM
3     print("Soal no 10")
4
5     index = 0
6     for i in self.NPM:
7         if (int(self.NPM[index])%2 != 0) & (int(self.NPM[index])
8             != 0):
9             print(self.NPM[index])
10            index += 1

```

Listing 2.31 Digit Ganjil NPM

2.41 Bilangan Prima NPM

```

1 def PrimaNPM(self):
2     #bilangan prima NPM
3     print("Soal no 11")
4     index = 0
5
6     for i in self.NPM:
7         prima = True
8         var=int(self.NPM[index])
9         if(var<=1):
10             prima=False
11         for i in range(2,var):
12             if(var%i==0):
13                 prima=False
14         if(prima==True):
15             print(var,"Prima")
16         else:
17             print(var,"bukan prima")
18         index += 1

```

Listing 2.32 Bilangan Prima NPM

2.42 Pemanggilan Fungsi pada Main.py

```

1 import fungsi
2
3 print("Masukkan NPM anda: ")
4 NPM = input()
5 crot = fungsi.Kelas(NPM)
6
7 soal1 = crot.Modulus()
8 soal2 = crot.Hello_NPM()
9 soal3 = crot.NPM_DigitBelakang()

```

```

10 soal14 = crot.NPM_DigitKetiga()
11 soal15 = crot.Variabel_Alfabet()
12 soal16 = crot.Penjumlahan_NPM()
13 soal17 = crot.Perkalian_NPM()
14 soal18 = crot.Print_Vertical()
15 soal19 = crot.DigitGenap()
16 soal10 = crot.DigitGanjil()
17 soal11 = crot.PrimaNPM()

```

Listing 2.33 Bilangan Prima NPM

2.43 Pengelolaan File CSV, Sejarah, dan Contoh

File csv (comma separated value) sering digunakan dalam dunia pemrograman untuk menampilkan data. file csv memiliki format yang sangat sederhana, setiap baris dipisahkan oleh enter dan setiap kolom dipisahkan oleh tanda koma.

Kegunaan file csv dibandingkan file dengan format lainnya adalah dari segi kompatibilitas karena file csv dapat diolah, dimodifikasi, digunakan, import/export menggunakan berbagai software dan bahasa pemrograman, salah satunya python.

Contoh file csv:

```

1 nomor , nama klub , jumlah main , poin , tanggal
2 1 , Manchester City , 8 , 19 , 1/2/1999
3 2 , Arsenal , 8 , 18 , 1/3/1999
4 3 , Tottenham Hotspurs , 8 , 18 , 1/4/1999
5 4 , Liverpool , 8 , 17 , 1/5/1999
6 5 , Chelsea , 8 , 16 , 1/6/1999
7 6 , Everton , 8 , 15 , 1/7/1999
8 7 , Manchester United , 8 , 14 , 1/8/1999
9 8 , Southampton , 8 , 12 , 1/9/1999
10 9 , AFC Bournemouth , 8 , 12 , 1/10/1999
11 10 , Crystal Palace , 8 , 11 , 1/11/1999

```

Listing 2.34 Contoh CSV

Contoh kode program python untuk membaca file csv.

```

1 #CSV Reader
2 import csv
3
4 with open('FCSV.csv') as csvfile:
5     readCSV = csv.reader(csvfile, delimiter=',')
6     for row in readCSV:

```

Listing 2.35 Contoh Kode Python to Read CSV

Contoh kode program python untuk membaca file csv menggunakan library pandas.

```

1 import pandas as pd
2 df1=pd.read_csv("FCSV.csv")
3 print(df1)

```

Listing 2.36 Contoh Kode Python to Read CSV

Sejarah CSV.

CSV sudah digunakan sejak tahun 1972, CSV dapat dikompilasi pada bahasa pemrograman IBM Fortran. Data yang dipisahkan oleh koma apabila terdapat spasi di dalamnya maka harus diberi tanda petik di awal dan akhir isi dari data tersebut. Nama CSV digunakan pada tahun 1983. Pada panduan dari Osborne Executive Computer terdapat kutipan yang membolehkan isi karakter memiliki koma. Pada tahun 2005 dengan RFC4180, CSV didefinisikan sebagai MIME Content Type. Lalu pada tahun 2013, defisiensi dari RFC4180 dipecahkan oleh rekomendasi dari W3C. Pada tahun 2014, IETF mempublikasi RFC7111 yang mendeskripsikan pecahan Uniform Resource Identifier(URI) ke dokumen CSV. RFC7111 menjelaskan tentang bagaimana baris, kolom dapat digunakan dalam dokumen CSV menggunakan indeks posisi. Pada Tahun 2015, W3C mempublikasikan draft rekomendasi untuk CSV-metadata standard yang dimulai pada bulan Desember 2015.

2.44 Aplikasi yang bisa Menciptakan File CSV

Aplikasi yang dapat kita gunakan untuk membuat file csv ada banyak, diantaranya:

1. Spreadsheet

Spreadsheet merupakan aplikasi pembuatan file CSV dengan cara memasukan data sesuai baris dan kolom yang diinginkan. Contoh spreadsheet seperti Google Spreadsheet, Microsoft Excel, dan aplikasi lainnya.

2. Bahasa

Pemrograman

Bahasa pemrograman merupakan media untuk membuat aplikasi yang dapat digunakan untuk membuat file CSV khusus dengan bahasa pemrograman tertentu yang support dengan pembuatan file CSV. Seperti Python, C Sharp, dan lain sebagainya.

3. Notepad

atau

Text

Editor

Text editor juga dapat membuat file CSV, cukup dengan membuat file sesuai format CSV dan save file tersebut dengan ekstensi .csv.

2.45 Menulis dan Membaca File CSV pada Ms.Excel

Membuat file csv melalui ms.excel sangatlah mudah, isikan nomor pada kolom, kemudian isikan variabel sebagai judul pada baris, lalu isikan data sesuai dengan variabel yang telah ditentukan, contoh:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	nomor	nama klub	jumlah main	poin																			
2	1	Manchester	8	19																			
3	2	Arsenal	8	18																			
4	3	Tottenham	8	18																			
5	4	Liverpool	8	17																			
6	5	Chelsea	8	16																			
7	6	Everton	8	15																			
8	7	Manchest	8	14																			
9	8	Southamp	8	12																			
10	9	AFC Bour	8	12																			
11	10	Crystal Pa	8	11																			
12																							
13																							
14																							
15																							
16																							
17																							
18																							
19																							
20																							
21																							
22																							
23																							
24																							
25																							
26																							
27																							
28																							
29																							
30																							

Gambar 2.13 Contoh Penulisan CSV pada Excel

Setelah mengetikkan data pada excel, lalu pilih menu file, pilih menu export, pilih menu change file type, pilih csv, lalu klik tombol save as. Maka file excel akan menjadi file csv.

```
1 nomor , nama klub , jumlah main , poin , tanggal
2 1 , Manchester ,8 ,19 ,1/2/1999
3 2 , Arsenal ,8 ,18 ,1/3/1999
4 3 , Tottenham Hotspurs ,8 ,18 ,1/4/1999
5 4 , Liverpool ,8 ,17 ,1/5/1999
6 5 , Chelsea ,8 ,16 ,1/6/1999
7 6 , Everton ,8 ,15 ,1/7/1999
8 7 , Manchester United ,8 ,14 ,1/8/1999
9 8 , Southampton ,8 ,12 ,1/9/1999
10 9 , AFC Bournemouth ,8 ,12 ,1/10/1999
11 10 , Crystal Palace ,8 ,11 ,1/11/1999
```

Listing 2.37 Contoh CSV

2.46 Sejarah Library CSV

Library CSV pada python merupakan library yang paling umum untuk import export data pada spreadsheet dan basis data dengan format sesuai dengan standarisasi RFC4180. Seiring dengan lahirnya bahasa pemrograman python, library mulai dibuat dan dikembangkan sampai akhirnya pada tahun 2003, Kevin Altis dan lainnya telah merilis versi final untuk library Python CSV.

2.47 Sejarah Library Pandas

Pandas (Python Data Analysis Library) adalah library open source yang digunakan untuk melakukan data manajemen dan data analysis. Pandas diciptakan pada tahun 2008 oleh Wes McKinney dan diperbaharui oleh Sien Chang pada tahun 2010. Inspirasi dari pembuatan pandas muncul pada komunitas yang membutuhkan library khusus untuk analisis data.

2.48 Fungsi - fungsi yang terdapat di library CSV

Berikut fungsi-fungsi yang terdapat pada library csv.

1. `csv.reader(csvfile, dialect='excel', **fmtparams)`

Untuk mengembalikan object reader yang akan mengambil setiap line pada csv. Data setiap baris diambil saat next() dipanggil. Berikut contohnya :

```
1 #CSV Reader
2 import csv
3
4 with open('FCSV.csv') as csvfile:
5     readCSV = csv.reader(csvfile, delimiter=',')
6     for row in readCSV:
7         print(row)
```

2. `csv.writer(csvfile, dialect='excel', **fmtparams)`

Mengembalikan file pembuat object untuk dapat mengkonversi data pada python ke file CSV yang akan dibuat. Berikut contoh penggunaan csv.writer :

```
1 #CSV Writer
2 import csv
3
4 with open('asal.csv', mode='w') as csvfile:
5     fieldnames = ['nomor', 'nama klub', 'jumlah main', 'poin']
6     csv_writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
7
8     csv_writer.writeheader()
9     csv_writer.writerow({'nomor': '1', 'nama klub': 'Asal-Asalan',
10                         'jumlah main': '8', 'poin': '10'})
11     csv_writer.writerow({'nomor': '2', 'nama klub': 'Asal-Asalan2',
12                         'jumlah main': '8', 'poin': '12'})
```

3. `csv.register_dialect(name[, dialect[, **fmtparams]])`

Mengasosiasikan dialek dengan nama, nama yang dimasukkan harus berupa karakter.

4. `csv.unregister_dialect(name)`

Menghapus asosiasi dialek dengan nama pada registry dialek.

5. `csv.get_dialect(name)`

Mengambil dialek yang telah diasosiasikan dengan nama.

6. `csv.list_dialects()`

Mengembalikan dialek yang telah diregistrasi.

7. `csv.field_size_limit([new_limit])`

Mengembalikan maksimal kolom data yang diperbolehkan oleh pembaca.

2.49 Fungsi - fungsi yang terdapat di library Pandas

1. `pandas.read_csv(filepath_or_buffer[, sep,])`

Untuk membaca file CSV dan menyimpannya ke DataFrame. Contohnya:

```
1 import pandas as pd
2 df1=pd.read_csv("FCSV.csv")
3 print(df1)
```

2. `pandas.read_excel(io[, sheet_name, header, names,])`

Membaca file excel dan menyimpannya ke DataFrame. Contohnya:

```
1 import pandas as pds
2 df2=pds.read_excel("FCSV.xlsx", index_col=None, header=None)
3 print(df2)
```

3. `to_csv([path, index, sep, na_rep,])`

Untuk membuat file CSV dari data yang ada

2.50 Praktek CSV

2.50.1 CSV List

```
1 import csv
2
3 def CsvList():
4     with open('FCSV.csv') as csv_file:
5         csv_reader = csv.reader(csv_file, delimiter=',')
6         line_count = 0
7         for row in csv_reader:
8             if line_count == 0:
9                 print(f'Nama Kolom adalah {", ".join(row)}')
10                line_count += 1
11            else:
12                print(f'\t Nomor : {row[0]}, Nama Klub : {row[1]},')
13                Jumlah Main : {row[2]}, Poin : {row[3]}.')
14                line_count += 1
15                print(f'Jumlah Total {line_count} lines.')
```

Listing 2.38 CSV List

2.50.2 CSV Dictionary

```

1 def CsvDict():
2     with open('dictionary.csv', mode='r') as csv_file:
3         csv_reader = csv.DictReader(csv_file)
4         line_count = 0
5         for row in csv_reader:
6             if line_count == 0:
7                 print(f'Nama Kolom adalah {", ".join(row)}')
8                 line_count += 1
9             else:
10                 print(f'\t First Name : {row["first_name"]} Last Name
11 : {row["last_name"]}.')
12                 line_count += 1
13                 print(f'Jumlah Total {line_count} lines.')

```

Listing 2.39 CSV Dictionary

2.50.3 Hello NPM (Pandas List)

```

1 import pandas
2
3 def pandasList():
4     pd = pandas.read_csv("FCSV.csv")
5     print(pd)

```

Listing 2.40 Pandas List

2.50.4 Hello NPM (Pandas Dictionary)

```

1 def pandasDict():
2     pds = pandas.read_csv("dictionary.csv")
3     crot = pandas.DataFrame.from_dict(pds)
4     print(crot)

```

Listing 2.41 Pandas Dictionary

2.50.5 Pandas Date

```

1 def Tanggal():
2     tanggal = pandas.read_csv("FCSV.csv")
3     tanggal['tanggal']=pandas.to_datetime(tanggal['tanggal'])
4     print(tanggal)

```

Listing 2.42 Pandas Date

2.50.6 Pandas Ubah Index

```

1 def ubahIndex():
2     baris = pandas.read_csv("FCSV.csv")
3     barisindex = ['1', '2', '0', '3', '5', '4', '6', '7', '9', '8']
4     oke = baris.reindex(barisindex)
5     print(oke)

```

Listing 2.43 Pandas Ubah Index

2.50.7 Pandas Ubah Column

```

1 def ubahColumn():
2     ubah = pandas.read_csv("FCSV.csv")
3     okey = ubah.rename(columns={"nomor" : "number"})
4     print(okey)

```

Listing 2.44 Pandas Ubah Column

2.50.8 Main CSV

```

1 import D1184011_csv
2
3 csvList = D1184011_csv.CsvList()
4 csvDict = D1184011_csv.CsvDict()

```

Listing 2.45 Main CSV

2.50.9 Main Pandas

```

1 import D1184011_pandas
2
3 pandasList = D1184011_pandas.pandasList()
4 pandasDict = D1184011_pandas.pandasDict()
5 pandasDate = D1184011_pandas.Tanggal()
6 pandasUbah = D1184011_pandas.ubahIndex()
7 pandasUbahCol = D1184011_pandas.ubahColumn()

```

Listing 2.46 Digit Genap NPM

2.51 *Speech to text*

Sistem konvensional yang dikenal untuk mengubah ucapan ke teks yang melibatkan pengenalan ucapan otomatis adalah sistem desktop stand alone, di mana setiap pengguna membutuhkan sistemnya sendiri. Sistem konversi bicara ke teks yang dikenal seperti itu telah diproduksi oleh perusahaan seperti Mesin Bisnis Internasional, Kurzweil Applied Intelligence Inc dan Dragon Systems. Sistem yang dikenal ini mampu mentranskripsikan ucapan manusia ke teks, meskipun tidak sempurna. Hasil teks disajikan kepada pengguna setelah penundaan kecil sementara dia masih mendikte.

sistem konversi ucapan-ke-teks yang terdiri dari setidaknya satu terminal pengguna untuk merekam ucapan, setidaknya satu prosesor pengenalan suara otomatis untuk menghasilkan teks dari file pidato yang direkam, dan komunikasi berarti operasi untuk mengembalikan file teks yang sesuai kepada pengguna.

2.52 Sejarah speech

Pengenalan ucapan atau pengenalan wicara dalam istilah bahasa Inggrisnya, automatic speech recognition (ASR) adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan. Teknologi ini memungkinkan suatu perangkat untuk mengenali dan memahami kata-kata yang diucapkan dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut dengan suatu pola tertentu yang tersimpan dalam suatu perangkat. Kata-kata yang diucapkan diubah bentuknya menjadi sinyal digital dengan cara mengubah gelombang suara menjadi sekumpulan angka yang kemudian disesuaikan dengan kode-kode tertentu untuk mengidentifikasi kata-kata tersebut. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi sebagai sebuah komando untuk melakukan suatu pekerjaan, misalnya penekanan tombol pada telepon genggam yang dilakukan secara otomatis dengan komando suara.

Alat pengenal ucapan, yang sering disebut dengan speech recognizer, membutuhkan sampel kata sebenarnya yang diucapkan dari pengguna. Sampel kata akan didigitalisasi, disimpan dalam komputer, dan kemudian digunakan sebagai basis data dalam mencocokkan kata yang diucapkan selanjutnya. Sebagian besar alat pengenal ucapan sifatnya masih tergantung kepada pembicara. Alat ini hanya dapat mengenal kata yang diucapkan dari satu atau dua orang saja dan hanya bisa mengenal kata-kata terpisah, yaitu kata-kata yang dalam penyampaiannya terdapat jeda antar kata. Hanya sebagian kecil dari peralatan yang menggunakan teknologi ini yang sifatnya tidak tergantung pada pembicara. Alat ini sudah dapat mengenal kata yang diucapkan oleh banyak orang dan juga dapat mengenal kata-kata kontinu, atau kata-kata yang dalam penyampaiannya tidak terdapat jeda antar kata.

Pengenalan ucapan dalam perkembangan teknologinya merupakan bagian dari pengenalan suara (proses identifikasi seseorang berdasarkan suaranya). Pengenalan suara sendiri terbagi menjadi dua, yaitu pengenalan pembicara (identifikasi suara berdasarkan orang yang berbicara) dan pengenalan ucapan (identifikasi suara berdasarkan kata yang diucapkan).

2.52.1 Perkembangan alat pengenal ucapan

Sejak tahun 1940, perusahaan American Telephone and Telegraph Company (ATT) sudah mulai mengembangkan suatu perangkat teknologi yang dapat mengidentifikasi kata yang diucapkan manusia. Sekitar tahun 1960-an, para peneliti dari perusahaan tersebut sudah berhasil membuat suatu perangkat yang dapat mengidentifikasi kata-kata terpisah dan pada tahun 1970-an mereka berhasil membuat perangkat yang

dapat mengidentifikasi kata-kata kontinu. Alat pengenal ucapan kemudian menjadi sangat fungsional sejak tahun 1980-an dan masih dikembangkan dan terus ditambahkan keefektifannya hingga sekarang.

2.52.2 Jenis-jenis pengenalan ucapan

Berdasarkan kemampuan dalam mengenal kata yang diucapkan, terdapat 5 jenis pengenalan kata, yaitu:

1. Kata-kata yang terisolasi

Proses pengidentifikasiannya hanya dapat mengenal kata yang diucapkan jika kata tersebut memiliki jeda waktu pengucapan antar kata

2. Kata-kata yang berhubungan

Proses pengidentifikasiannya mirip dengan kata-kata terisolasi, namun membutuhkan jeda waktu pengucapan antar kata yang lebih singkat

3. Kata-kata yang berkelanjutan

Proses pengidentifikasiannya sudah lebih maju karena dapat mengenal kata-kata yang diucapkan secara berkesinambungan dengan jeda waktu yang sangat sedikit atau tanpa jeda waktu. Proses pengenalan suara ini sangat rumit karena membutuhkan metode khusus untuk membedakan kata-kata yang diucapkan tanpa jeda waktu. Pengguna perangkat ini dapat mengucapkan kata-kata secara natural

4. Kata-kata yang berkelanjutan

Proses pengidentifikasiannya sudah lebih maju karena dapat mengenal kata-kata yang diucapkan secara berkesinambungan dengan jeda waktu yang sangat sedikit atau tanpa jeda waktu. Proses pengenalan suara ini sangat rumit karena membutuhkan metode khusus untuk membedakan kata-kata yang diucapkan tanpa jeda waktu. Pengguna perangkat ini dapat mengucapkan kata-kata secara natural

5. Verifikasi atau identifikasi suara

Proses pengidentifikasiannya tidak hanya mampu mengenal kata, namun juga mengidentifikasi siapa yang berbicara.

2.53 Proses kerja alat pengenal ucapan

1. Tahap penerimaan masukan

Masukan berupa kata-kata yang diucapkan lewat pengeras suara.

2. Tahap ekstraksi

Tahap ini adalah tahap penyimpanaan masukan yang berupa suara sekaligus pembuatan basis data sebagai pola. Proses ekstraksi dilakukan berdasarkan metode Model Markov Tersembunyi atau Hidden Markov Model (HMM), yang merupakan model statistik dari sebuah sistem yang diasumsikan oleh Markov sebagai suatu proses dengan parameter yang tidak diketahui. Tantangan dalam model statistik ini adalah menentukan parameter-parameter tersembunyi dari parameter yang dapat diamati. Parameter-parameter yang telah kita tentukan kemudian digunakan untuk analisis yang lebih jauh pada proses pengenalan kata yang diucapkan. Berdasarkan HMM, proses pengenalan ucapan secara umum menghasilkan keluaran yang dapat dikarakterisasikan sebagai sinyal. Sinyal dapat bersifat diskrit (karakter dalam abjad) maupun kontinu (pengukuran temperatur, alunan musik).

Sinyal dapat pula bersifat stabil (nilai statistiknya tidak berubah terhadap waktu) maupun nonstabil (nilai sinyal berubah-ubah terhadap waktu). Dengan melakukan pemodelan terhadap sinyal secara benar, dapat dilakukan simulasi terhadap masukan dan pelatihan sebanyak mungkin melalui proses simulasi tersebut sehingga model dapat diterapkan dalam sistem prediksi, sistem pengenalan, maupun sistem identifikasi. Secara garis besar model sinyal dapat dikategorikan menjadi dua golongan, yaitu: model deterministik dan model statistikal.

Model deterministik menggunakan nilai-nilai properti dari sebuah sinyal seperti: amplitudo, frekuensi, dan fase dari gelombang sinus. Model statistikal menggunakan nilai-nilai statistik dari sebuah sinyal seperti: proses Gaussian, proses Poisson, proses Markov, dan proses Markov Tersembunyi. Suatu model HMM secara umum memiliki unsur-unsur sebagai berikut:

- (a) N, yaitu jumlah bagian dalam model. Secara umum bagian tersebut saling terhubung satu dengan yang lain, dan suatu bagian bisa mencapai semua bagian yang lain, serta sebaliknya (disebut dengan model ergodik). Namun hal tersebut tidak mutlak karena terdapat kondisi lain dimana suatu bagian hanya bisa berputar ke diri sendiri dan berpindah ke satu bagian berikutnya. Hal ini bergantung pada implementasi dari model.
- (b) M, yaitu jumlah simbol observasi secara unik pada tiap bagianya, misalnya: karakter dalam abjad, dimana bagian diartikan sebagai huruf dalam kata.
- (c) M, yaitu jumlah simbol observasi secara unik pada tiap bagianya, misalnya: karakter dalam abjad, dimana bagian diartikan sebagai huruf dalam kata.
- (d) Probabilitas Simbol Observasi pada bagian j, $P(j|Bb_k)$

(e) Inisial Distribusi Bagian i p p

Setelah memberikan nilai N, M, A, B, dan p, maka proses ekstraksi dapat diurutkan. Berikut adalah tahapan ekstraksi pengenalan ucapan berdasarkan HMM:

(a) Tahap ekstraksi tampilan

Penyaringan sinyal suara dan pengubahan sinyal suara analog ke digital

(b) Tahap tugas pemodelan

Pembuatan suatu model HMM dari data-data yang berupa sampel ucapan sebuah kata yang sudah berupa data digital

(c) Tahap sistem pengenalan HMM

Penemuan parameter-parameter yang dapat merepresentasikan sinyal suara untuk analisis lebih lanjut.

3. Tahap pembandingan

Tahap ini merupakan tahap pencocokan data baru dengan data suara (pencocokan tata bahasa) pada pola. Tahap ini dimulai dengan proses konversi sinyal suara digital hasil dari proses ekstraksi ke dalam bentuk spektrum suara yang akan dianalisis dengan membandingkannya dengan pola suara pada basis data. Sebelumnya, data suara masukan dipilah-pilah dan diproses satu per satu berdasarkan urutannya.

Pemilihan ini dilakukan agar proses analisis dapat dilakukan secara paralel. Proses yang pertama kali dilakukan ialah memproses gelombang kontinu spektrum suara ke dalam bentuk diskrit. Langkah berikutnya ialah proses kalkulasi yang dibagi menjadi dua bagian:

(a) Transformasi gelombang diskrit menjadi data yang terurut Gelombang diskrit berbentuk masukan berukuran n yang menjadi objek yang akan dibagi pada proses konversi dengan cara pembagian rincian waktu

(b) Menghitung frekuensi pada tiap elemen data yang terurut

Selanjutnya tiap elemen dari data yang terurut tersebut dikonversi ke dalam bentuk bilangan biner. Data biner tersebut nantinya akan dibandingkan dengan pola data suara dan kemudian diterjemahkan sebagai keluaran yang dapat berbentuk tulisan ataupun perintah pada perangkat.

4. Tahap validasi identitas pengguna

Alat pengenal ucapan yang sudah memiliki sistem verifikasi/identifikasi suara akan melakukan identifikasi orang yang berbicara berdasarkan kata yang diucapkan setelah menerjemahkan suara tersebut menjadi tulisan atau komando.

2.54 Aplikasi alat pengenal ucapan

2.54.1 Bidang komunikasi

Perintah Suara

Perintah Suara (komando suara) adalah suatu program pada komputer yang melakukan perintah berdasarkan perintah suara dari pengguna. Contohnya pada aplikasi Microsoft Voice yang berbasis bahasa Inggris. Ketika pengguna mengatakan Mulai kalkulator dengan intonasi dan tata bahasa yang sesuai, komputer akan segera membuka aplikasi kalkulator. Jika komando suara yang diberikan sesuai dengan daftar perintah yang tersedia, aplikasi akan memastikan komando suara dengan menampilkan tulisan Apakah Anda meminta saya untuk mulai kalkulator?. Untuk melakukan verifikasi, pengguna cukup mengatakan Lakukan dan komputer akan langsung beroperasi.

Pendiktean

Pendiktean adalah sebuah proses mendikte yang sekarang ini banyak dimanfaatkan dalam pembuatan laporan atau penelitian. Contohnya pada aplikasi Microsoft Dictation yang merupakan aplikasi yang dapat menuliskan apa yang diucapkan oleh pengguna secara otomatis.

Telepon

Pada telepon, teknologi pengenal ucapan digunakan pada proses penekanan tombol otomatis yang dapat menelpon nomor tujuan dengan komando suara.

2.54.2 Bidang kesehatan

Alat pengenal ucapan banyak digunakan dalam bidang kesehatan untuk membantu para penyandang cacat dalam beraktivitas. Contohnya pada aplikasi Antarmuka Suara Pengguna atau Voice User Interface (VUI) yang menggunakan teknologi pengenal ucapan dimana pengendalian saklar lampu misalnya, tidak perlu dilakukan secara manual dengan menggerakkan saklar tetapi cukup dengan mengeluarkan perintah dalam bentuk ucapan sebagai saklarnya. Metode ini membantu manusia yang secara fisik tidak dapat menggerakkan saklar karena cacat pada tangan misalnya. Penyerapan VUI ini tidak hanya untuk lampu saja tapi bisa juga untuk aplikasi-aplikasi kontrol yang lain.

2.54.3 Bidang militer

Pelatihan Penerbangan

Aplikasi alat pengenal ucapan dalam bidang militer adalah pada pengatur lalu-lintas udara atau yang dikenal dengan Air Traffic Controllers (ATC) yang dipakai oleh para pilot untuk mendapatkan keterangan mengenai keadaan lalu-lintas udara seperti radar, cuaca, dan navigasi. Alat pengenal ucapan digunakan sebagai pengganti operator yang memberikan informasi kepada pilot dengan cara berdialog.

Helikopter

Aplikasi alat pengenal ucapan pada helikopter digunakan untuk berkomunikasi lewat radio dan menyesuaikan sistem navigasi. Alat ini sangat diperlukan pada helikopter karena ketika terbang, sangat banyak gangguan yang akan menyulitkan pilot bila harus berkomunikasi dan menyesuaikan navigasi dengan terlebih dahulu memencet tombol tertentu.

2.55 Kelebihan alat pengenal ucapan

Kelebihan dari peralatan yang menggunakan teknologi ini adalah:

1. Cepat

Teknologi ini mempercepat transmisi informasi dan umpan balik dari transmisi tersebut. Contohnya pada komando suara. Hanya dalam selang waktu sekitar satu atau dua detik setelah kita mengkomandokan perintah melalui suara, komputer sudah memberi umpan balik atas komando kita.

2. Mudah digunakan

Kemudahan teknologi ini juga dapat dilihat dalam aplikasi komando suara. Komando yang biasanya kita masukkan ke dalam komputer dengan menggunakan tetikus atau papan ketik kini dapat dengan mudahnya kita lakukan tanpa perangkat keras, yakni dengan komando suara.

2.56 Kekurangan alat pengenal ucapan

1. Rawan terhadap gangguan

Hal ini disebabkan oleh proses sinyal suara yang masih berbasis frekuensi. Ketika sebuah informasi dalam sinyal suara mempunyai komponen frekuensi yang sama banyaknya dengan komponen frekuensi gangguannya, akan sulit untuk memisahkan gangguan dari sinyal suara

2. Jumlah kata yang dapat dikenal terbatas

Hal ini disebabkan pengenal ucapan bekerja dengan cara mencari kemiripan dengan basis data yang dimiliki.

2.57 speech recognition

Sebelum kita sampai pada seluk-beluk melakukan pengenalan suara dengan Python, mari kita luangkan waktu untuk berbicara tentang cara kerja pengenalan suara. Pengenalan ucapan berakar pada penelitian yang dilakukan di Bell Labs pada awal 1950-an. Sistem awal terbatas pada satu pembicara dan memiliki kosakata terbatas sekitar selusin kata. Sistem pengenalan ucapan modern telah berkembang sejak rekan-rekan kuno mereka. Mereka dapat mengenali pembicaraan dari banyak penutur dan memiliki banyak sekali kosakata dalam berbagai bahasa. Komponen pertama dari pengenalan ucapan adalah, tentu saja, ucapan. Pidato harus dikonversi dari suara fisik ke sinyal listrik dengan mikrofon, dan kemudian ke data digital dengan konverter analog-ke-digital.

Setelah didigitalkan, beberapa model dapat digunakan untuk menyalin audio ke teks. Sebagian besar sistem pengenalan ucapan modern mengandalkan apa yang dikenal sebagai Hidden Markov Model (HMM). Pendekatan ini bekerja berdasarkan asumsi bahwa sinyal wicara, bila dilihat pada skala waktu yang cukup singkat (katakanlah, sepuluh milidetik), dapat diperkirakan secara wajar sebagai proses stasioner yaitu, proses di mana sifat statistik tidak berubah seiring waktu.

Dalam tipikal HMM, sinyal ucapan dibagi menjadi fragmen 10 milidetik. Spektrum daya dari setiap fragmen, yang pada dasarnya adalah sebidang kekuatan sinyal sebagai fungsi frekuensi, dipetakan ke vektor bilangan real yang dikenal sebagai koefisien cepstral . Dimensi vektor ini biasanya kecil kadang-kadang serendah 10, meskipun sistem yang lebih akurat mungkin memiliki dimensi 32 atau lebih. Hasil akhir dari HMM adalah urutan vektor-vektor ini.

Orang dapat membayangkan bahwa keseluruhan proses ini mungkin mahal secara komputasi. Dalam banyak sistem pengenalan ucapan modern, jaringan saraf digunakan untuk menyederhanakan sinyal ucapan menggunakan teknik untuk transformasi fitur dan pengurangan dimensi sebelum pengakuan HMM. Detektor aktivitas suara (VAD) juga digunakan untuk mengurangi sinyal audio menjadi hanya bagian-bagian yang cenderung mengandung ucapan. Ini mencegah pengenal dari membuang-buang waktu menganalisis bagian yang tidak perlu dari sinyal.

Untungnya, sebagai programmer Python, Anda tidak perlu khawatir tentang semua ini. Sejumlah layanan pengenalan ucapan tersedia untuk digunakan online melalui API, dan banyak dari layanan ini menawarkan Python SDK .

2.57.1 Paket Pengenalan Python Speech

1. apiai
2. assemblyai
3. google-cloud-speech
4. pocketsphinx
5. SpeechRecognition
6. watson-developer-cloud
7. wit

2.58 command line interface

Antarmuka baris perintah (bahasa Inggris: command-line interface, CLI) adalah mekanisme interaksi dengan sistem operasi atau perangkat lunak komputer dengan mengetikkan perintah untuk menjalankan tugas tertentu. Antarmuka hanya-teks ini merupakan kontras dari penggunaan peranti penunjuk untuk mengeklik pilihan pada antarmuka pengguna grafis (GUI), atau penggunaan menu untuk memilih pilihan pada antarmuka pengguna teks (TUI). Konsep CLI dimulai sewaktu teletype-writer machine (TTY) dihubungkan ke komputer pada dasawarsa 1950-an dan terus berkembang bersama dengan sistem GUI seperti Microsoft Windows, Mac OS, dan X Window System. Pada beberapa aplikasi, seperti MATLAB dan AutoCAD, CLI terintegrasi dengan GUI dan mendapat manfaat dari keduanya.

sistem tertanam adalah komputer khusus tempat sistem operasi dan fungsi aplikasi sering digabungkan ke dalam program yang sama. Secara umum, sistem tertanam menyiratkan serangkaian fungsi tetap yang diprogram ke dalam memori non-volatile (ROM, flash memory, dll.) Berbeda dengan mesin komputasi tujuan umum.

Sistem yang disematkan semakin lama semakin kompleks, membutuhkan serangkaian aplikasi manajemen eksternal seperti SNMP, Command Line Interfaces, TNM, WBEM, dll. Aplikasi ini memiliki antarmuka yang sangat berbeda, protokol dan tujuan jaringan tetapi semua harus terhubung ke level rendah internal sistem tertanam.

Agar sistem tertanam dapat berfungsi dengan sukses di lingkungan, sistem tertanam harus dikonfigurasi untuk beroperasi sesuai keinginan untuk tujuan pemecahan masalah dan pemantauan. Selanjutnya, sistem yang disematkan harus dapat memberikan konfigurasi saat ini atau mengembalikan ke konfigurasi yang disimpan saat diminta. Fungsi ini secara kolektif dikenal sebagai "manajemen" dari sistem embedded.

2.59

Sejarah Selenium

Kisah ini dimulai pada 2004 di ThoughtWorks di Chicago, dengan Jason Huggins membangun mode Inti sebagai JavaScriptTestRunner untuk pengujian aplikasi waktu dan Pengeluaran internal (Python, Plone). Pengujian otomatis terhadap aplikasi apa pun adalah inti dari gaya ThoughtWork, mengingat kecenderungan Agile dari konsultasi ini. Dia mendapat bantuan dari Paul Gross dan Jie Tina Wang. Bagi mereka, ini adalah pekerjaan harian.

Jason mulai memperagakan alat uji ke berbagai rekan. Banyak yang senang dengan umpan balik visual yang langsung dan intuitif, serta potensinya untuk tumbuh sebagai kerangka pengujian yang dapat digunakan kembali untuk aplikasi web lainnya.

Segara setelah tahun 2004 sesama ThoughtWorker Paul Hammant melihat demo, dan memulai diskusi tentang sumber terbuka Selenium, serta mendefinisikan mode 'didorong' Selenium di mana Anda bisa menggunakan Selenium melalui kabel dari bahasa pilihan Anda. , yang akan menyiasati 'kebijakan asal yang sama'. Rekan-rekan (saat itu) lainnya, Aslak Hellesoy dan Mike Melia, bereksperimen dengan berbagai ide untuk karya 'server', termasuk penulisan ulang halaman untuk mendapatkan kebijakan asal yang sama. Paul menulis karya server asli di Jawa, dan Aslak dan Obie Fernandez mengangkut driver klien ke Ruby, menetapkan fondasi untuk driver dalam lebih banyak bahasa.

Pekerja Pikir di berbagai kantor di seluruh dunia mengambil Selenium untuk proyek komersial, dan berkontribusi kembali ke Selenium dari pelajaran yang dipetik dari proyek ini. Mike Williams, Darrell Deboer, dan Darren Cotterill semuanya membantu meningkatkan kemampuan dan ketahanannya.

2.60 Jenis-Jenis *Selenium*

Selenium adalah perangkat lunak yang berfungsi untuk mendukung pengembangan otomatisasi uji berbasis web aplikasi. Selenium menyediakan pengujian khusus terhadap domain bahasa, untuk melakukan tes menulis pada beberapa bahasa pemrograman yang populer, termasuk C, Groovy, Java, Pearl, PHP, Python, Ruby, dan juga Scala. Pengujian dapat berjalan melalui browser web apa saja dan dapat dilakukan melalui Sistem Operasi di Windows, Linux, dan Platform OS X.

Selenium Python Bindings menyediakan API yang sederhana untuk menulis uji fungsional menggunakan Selenium WebDriver, dan juga dapat mengakses semua fungsi Selenium WebDriver secara intuitif. Selenium Python Bindings menyediakan API yang cukup nyaman untuk melakukan suatu akses Selenium WebDrivers seperti di Firefox, Internet Explorer, Chrome, dll

Jenis-Jenis Selenium Sebagai Berikut :

1. *IDE selenium*

Selenium IDE (Lingkungan Pengembangan Terpadu) adalah sumber terbuka alat rekam dan putar untuk menghasilkan skrip Selenium, yang terintegrasi dengan browser web Firefox sebagai ekstensi. Ini adalah tes UI berbasis web yang terkenal alat otomatisasi yang mengekstrak segala jenis locator dari halaman web. Ada banyak yang bisa baik berbasis atribut atau berbasis struktur, dan termasuk ID, nama, tautan, XPath, CSS, dan DOM. IDE memiliki seluruh Selenium Core, yang memungkinkan pengguna mencatat 10, memutar, mengedit, dan men-debug tes secara manual di browser.

Tindakan pengguna di web halaman dapat direkam dan diekspor dalam bahasa apa saja yang paling populer, seperti Java, C , Ruby, dan Python, Selenium Builder adalah alat open source alternatif untuk dicatat oleh Selenium IDE dan pemutaran aplikasi web. Ini adalah ekstensi dari web browser Firefox, Yang mirip dengan Selenium IDE, tetapi, ia memiliki beberapa fitur unik yaitu Selenium IDE tidak mendukung. Selenium Builder adalah alat standar dari Sauce Labs yang menjalankan tes Sauce Cloud dari antarmuka Selenium Builder itu sendiri.

2. Selenium WebDriver

Selenium webdriver adalah versi terbaru dari selenium IDE dan selenium Remote Control (RC). Ini juga dinamai selenium 2.0. Ini memungkinkan skrip uji yang dirancang untuk berkomunikasi dengan browser secara langsung dengan bantuan metode asli. Ini mendukung pengujian aplikasi web pada desktop serta pada perangkat seluler seperti Android dan iOS perangkat.

Biaya proyek berkurang dengan bantuan ini alat karena itu adalah alat open-source. Waktu yang diperlukan untuk mengeksekusi skrip pengujian di webdriver kurang jika dibandingkan ke selenium IDE dan Selenium RC. Ini memungkinkan skrip pengujian dirancang untuk berbagai browser seperti Internet explorer, Firefox, Mac safari dan Chrome. Script pengujian dapat dikembangkan menggunakan bahasa seperti Java, C , Ruby, Perl, Python.

3. Remote Control Selenium

Remote Control Selenium (RC) adalah selenium utama yang digunakan untuk memproyeksikan waktu yang lama. Selenium RC lebih lambat daripada selenium webdriver karena menggunakan program java script yang disebut sebagai suatu inti dari selenium. Selenium RC harus memulai server sebelum menjalankan suatu skrip pengujian, dan itu tidak mendukung untuk aplikasi Ajax. Cara menghindari keterbatasan Selenium RC, atau dengan selenium Web Driver.

4. Selenium Grid

Server yang memungkinkan pengujian untuk menggunakan instance browser web yang sedang berjalan di mesin jarak jauh. Dengan selenium grid, satu

server bertindak sebagai hub. Tes hubungi hub untuk mendapatkan akses ke instance browser karena hub memiliki daftar server yang menyediakan akses ke instance browser (node WebDriver), dan memungkinkan pengujian menggunakan instance ini.

Selenium Grid memiliki kemampuan untuk menjalankan tes pada instance browser jarak jauh yang berguna untuk menyebarluaskan beban pengujian di beberapa mesin, dan untuk menjalankan tes di browser yang berjalan pada platform atau sistem operasi yang berbeda. Yang terakhir ini sangat berguna dalam kasus di mana tidak semua browser yang akan digunakan untuk pengujian dapat berjalan pada platform yang sama.

5. *TestNG*

TestNG adalah kerangka pengujian yang digunakan untuk pengujian otomatisasi bersama dengan selenium 2.0. Ini mendukung berbagai tingkat pengujian seperti unit, integrasi, sistem dan pengujian penerimaan pengguna (UAT). Biasanya disebut sebagai "Uji Generasi Baru".

2.61 Anaconda

Distribusi open-source Anaconda adalah cara termudah untuk melakukan sains data Python / R dan pembelajaran mesin di Linux, Windows, dan Mac OS X. Dengan lebih dari 15 juta pengguna di seluruh dunia, ini adalah standar industri untuk pengembangan, pengujian, dan pelatihan tentang mesin tunggal,

2.61.1 Instalasi Anaconda

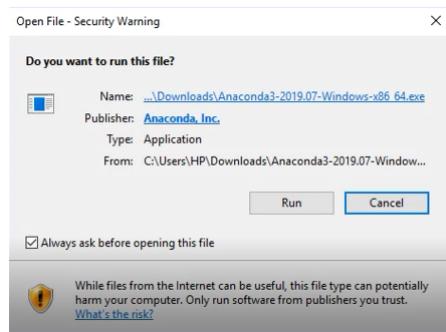
Hal yang harus diperhatikan sebelum melakukan instalasi *Anaconda Python*

1. Perhatikan versi dari sistem operasi yang digunakan (versi 32bit atau 64bit)
2. Download file anaconda yang sesuai dengan versi sistem operasi (32bit atau 64bit)

3. Download Anaconda Python <https://www.anaconda.com/distribution/>

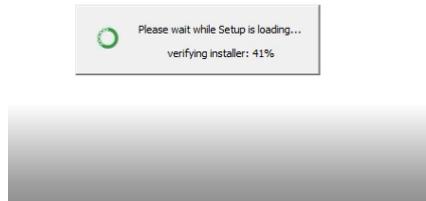
Berikut langkah-langkah instalasi anaconda.

1. Buka aplikasi *installer Anaconda* tersebut lalu akan muncul gambar *installer anaconda*.



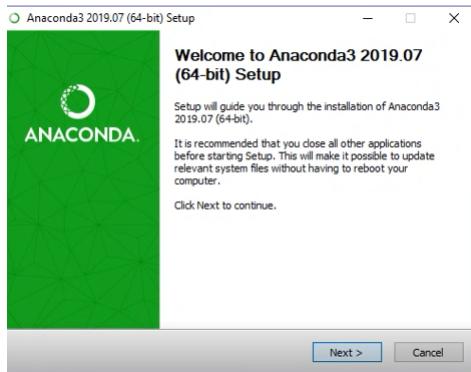
Gambar 2.14 Run Setup Anaconda

2. Tunggu hingga *setup loading* selesai



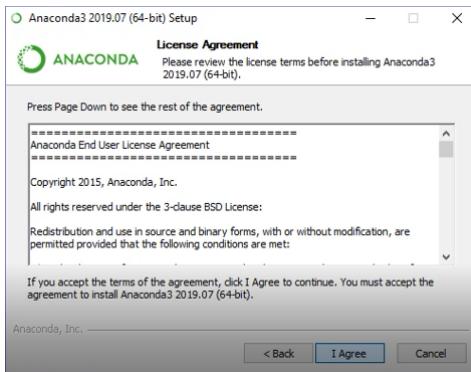
Gambar 2.15 Setup Loading

3. Jika *setup loading* telah selesai, maka klik *next*



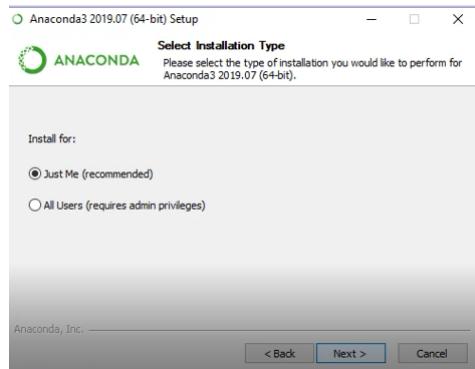
Gambar 2.16 Welcome to Anaconda Setup

4. Pada *License Agreement* klik *I Agree* gambar *License Agreement*.



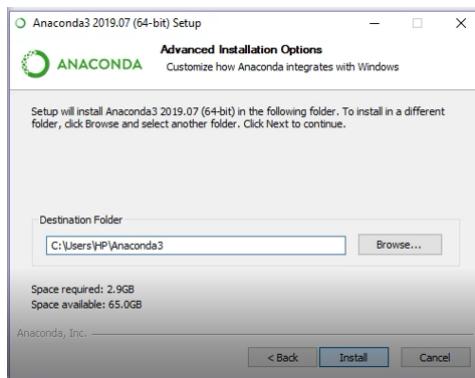
Gambar 2.17 License Agreement

5. Kemudian pilih *Just Me(Recomended)* agar sesuai dengan komputer yang digunakan, kemudian klik *next* gambar *Just Me(recomended)*.



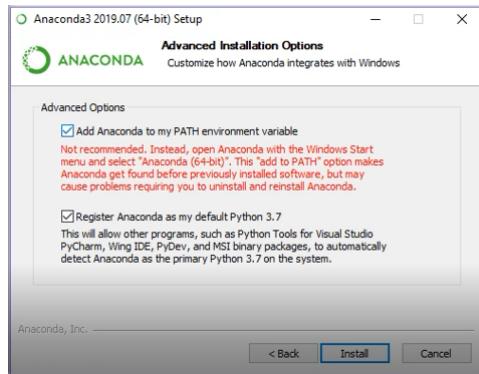
Gambar 2.18 Just Me(recomended)

6. Kemudian pilih lokasi tempat *menginstall anaconda* gambar *Pilih lokasi*.



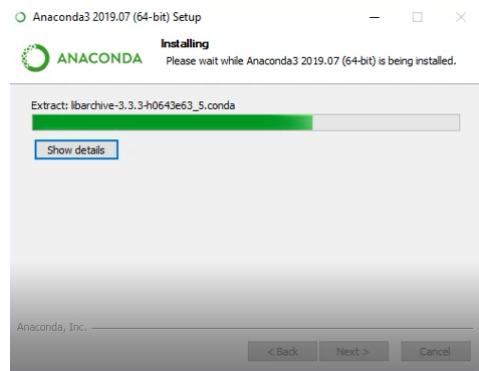
Gambar 2.19 Pilih lokasi

7. Kemudian centang *Add Anaconda to my Path environment variable*, agar saat *menginstall selenium* langsung ke *path anaconda* tidak ke aplikasi yang lain. Klik *install* gambar *Centang Anaconda to my PATH*.



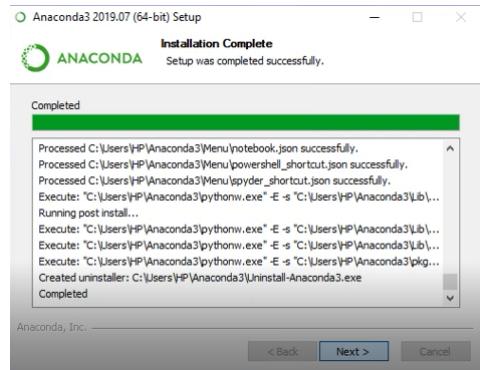
Gambar 2.20 Centang Anaconda to my PATH

8. Tunggu sampai proses *installasi* selesai gambar *Installation Complete*.



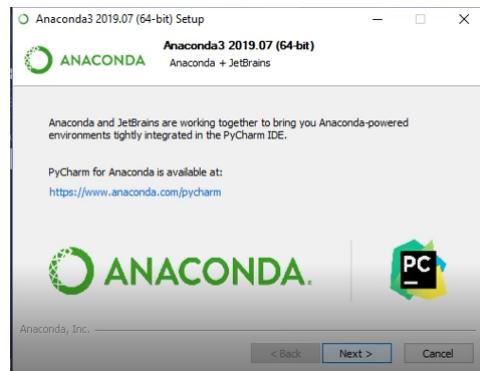
Gambar 2.21 Installation Complete

9. Apabila instalasi telah selesai klik *next*



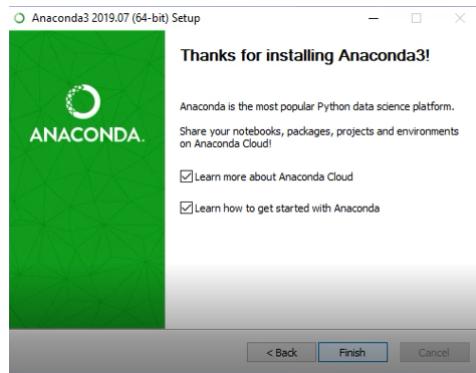
Gambar 2.22 Installation Complete

10. klik *next*



Gambar 2.23 Anaconda+JetBrains

11. Jika sudah klik *finish* gambar *Thanks fo install Anaconda.*



Gambar 2.24 *Thanks for install Anaconda*

2.62 Instalasi Pip

1. buka anaconda prompt

2. ketikkan Pip install selenium

(2).pdf (2).png (2).jpg (2).mps (2).jpeg (2).jbig2 (2).jb2 (2).PDF (2).PNG (2).JPG (2).JPEG (2)

Gambar 2.25 *Install pip*

3. Tunggu hingga proses instalasi selesai.

(2).pdf (2).png (2).jpg (2).mps (2).jpeg (2).jbig2 (2).jb2 (2).PDF (2).PNG (2).JPG (2).JPEG (2)

Gambar 2.26 *Install pip Selesai*

4. jika telah selesai, lakukan pengecekan versi pip dengan mengetikkan pip -V

(2).pdf (2).png (2).jpg (2).mps (2).jpeg (2).jbig2 (2).jb2 (2).PDF (2).PNG (2).JPG (2).JPEG (2)

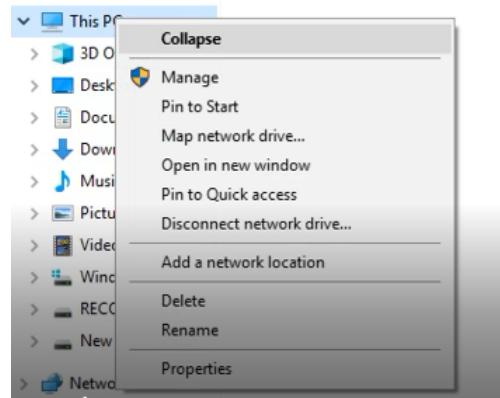
Gambar 2.27 *Melihat Versi pip*

2.63 Setting Environment

2.63.1 Windows (Windows 10)

1. Buka file explorer

2. Klik kanan pada This pc, lalu pilih properties



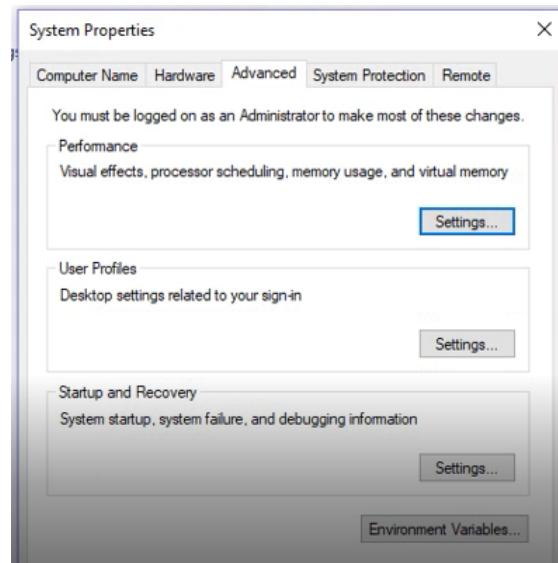
Gambar 2.28 *Properties*

3. Pilih menu Advanced system settings



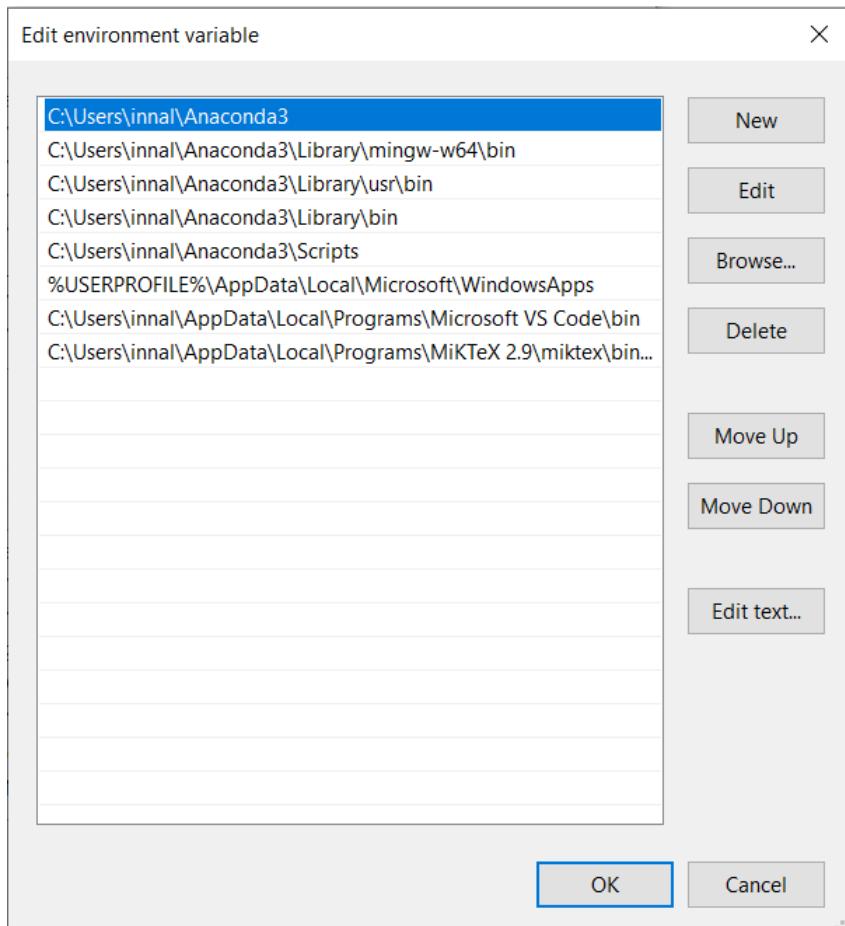
Gambar 2.29 *Advanced system settings*

4. Pilih Environment Variables



Gambar 2.30 *Environment Variables*

5. Pilih Path, lalu pilih environment variable yang ingin ditambahkan, klik OK



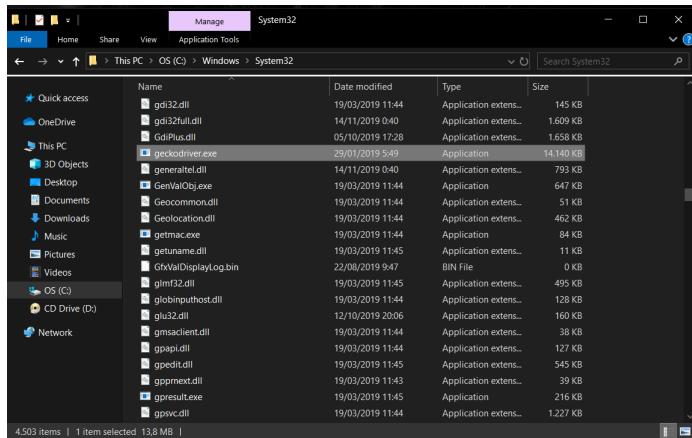
Gambar 2.31 Path

2.64 Geckodriver dan Chromedriver

2.64.1 Geckodriver untuk Mozilla Firefox

Download Geckodriver pada link ini <https://github.com/mozilla/geckodriver/releases>. sebelum mendownload harus menyamakan versi mozilla dengan versi Geckodriver misal versi Mozilla firefox versi 32bit Geckodrivernya pun harus 32bit jika tidak maka akan terjadi kesalahan.

jika sudah mendownload Geckodriver pindahkan file tersebut ke system32



Gambar 2.32 Gechodriver

BAB 3

PEMBANGUNAN APLIKASI

3.1 Pendahuluan

Sebelum membangun aplikasi ada beberapa hal yang di perlukan sebagai berikut :

1. Visual Studio Code

Editor kode sumber yang dikembangkan oleh Microsoft untuk Windows, Linux dan macOS. Ini mencakup dukungan untuk debugging, kontrol Git yang ter-tanam dan GitHub, penyorotan sintaksis, penyelesaian kode cerdas, cuplikan, dan refactoring kode.

2. Menginstall python

karena disini kami akan memakai bahasa pemrograman python jadi wajib menginstall python di komputer atau laptop agar bisa menggunakan bahasa pemrograman python untuk membuat sebuah aplikasi.

3. pip install SpeechRecognition module untuk *speech recognition*

4. pip install selenium module untuk selenium

Setelah mempersiapkan hal yang diperlukan mari kita memulai untuk membangun aplikasi Musicroot ini.

Pertama-tama import library *speech recognition*

```
| import speech_recognition as sr
```

sr tersebut adalah alias untuk speech_recognition agar saat ingin memanggil speech_recognition tidak usah repot-repot cukup memanggil sr saja karena itu sudah mewakili speech_recognition.

method

```
1 def assistant_speaks(output):
2     global num
3     num = 1
4     num += 1
5     print("Croot : ", output)
6     toSpeak = gTTS(text = output, lang = 'id', slow = False)
7     file = str(num)+".mp3"
8     toSpeak.save(file)
9     playsound.playsound(file, True)
10    os.remove(file)
```

method

method

```
1 def get_audio():
2     rObject = sr.Recognizer()
3     audio = ''
4     with sr.Microphone() as source:
5         print("Bicara . . .")
6         audio = rObject.listen(source, phrase_time_limit = 5)
7         print("Stop . . .")
8     try:
9
10        text = rObject.recognize_google(audio, language ='id-ID')
11        print("Anda : ", text)
12        return text
13
14    except:
15
16        assistant_speaks("tidak jelas")
17        return 0
```

method

method

```
1 if __name__ == "__main__":
2
3     assistant_speaks("Nama Anda Siapa ?")
4     name ='Manusia',
5     name = get_audio()
6     assistant_speaks("Hallo, " + name + ", ")
7
8     while(1):
9
10        assistant_speaks("Apa yang saya bisa bantu?")
11        text = get_audio().lower()
```

method

method

```
1 if "musik" in str(text) :  
2     assistant_speaks("Ok , "+ name+' Membuka mundur alon-alon')  
3     options = webdriver.ChromeOptions()  
4     driver = webdriver.Chrome(chrome_options=options)           driver.  
5     get('https://www.youtube.com/watch?v=zCmHodTpt9I')  
    continue
```

method

method

```
1 if "lyrics" in str(text) :  
2     assistant_speaks("Ok , "+ name+' Membuka lyrics mundur alon-alon'  
3     )  
4     options = webdriver.ChromeOptions()  
5     driver = webdriver.Chrome(chrome_options=options)  
6     driver.get('https://www.google.com/search?safe=strict&hl=id&sxsrf=  
=ACYBGNQDWNQ0tkOaPmxtoL29VOHZfkTEdw%3A1573181655017&source=hp&ei=  
=1tjExefEO4eEvQSLj4KwBQ&q=lirik+mundur+alon+alon&oq=l&gs_l=psy-ab  
.3.0.35 i3912j0i67j0i13j0i67j0i131j0i6712  
.26634.26634..28264...0.0..0.705.875.0j1j6 - 1.....0....1..gws-wiz  
.....10..35 i362i39 .oUYCZ9a37bg')  
    continue
```

method

method

```
1 if "chrome" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka google chrome')
3     os.startfile("C:/Program Files (x86)/Google/Chrome/Application/
4         chrome.exe")
        continue
```

method

method

```
1 if "spotify" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka spotify')
3     os.startfile ("C:/ Users/innal/AppData/Roaming/Spotify/Spotify.exe")
4     continue
```

method

method

```
1 if "last" in str(text) :  
2     assistant_speaks("Ok , "+ name+ ' Membuka last.fm')  
3     options = webdriver.ChromeOptions()  
4     driver = webdriver.Chrome(chrome_options=options)  
5     driver.get('https://www.last.fm/')6     driver.get('https://secure.last.fm/login')  
7     driver.find_element_by_name('username').send_keys(  
8         muchamadinnalk@gmail.com')  
9     driver.find_element_by_xpath('//*[@[@id="id_password"]]').send_keys(  
10        'samsung123!')  
11     driver.get("https://www.last.fm/search")  
12     driver.find_element_by_link_text('Search').send_keys("noah")  
13     driver.find_element_by_class_name('search-submit').click()  
14     driver.get('https://www.last.fm/music/Noah')  
15     continue
```

method

method

```
1 if "download" in str(text) :  
2     assistant_speaks("Ok , "+ name+' Membuka last.fm')  
3     options = webdriver.ChromeOptions()  
4     driver = webdriver.Chrome(chrome_options=options)  
5     driver.get("https://www.google.com/")  
6     driver.find_element_by_css_selector("#tsf > div:nth-child(2) >  
7         div.A8SBwf > div.RNNXgb > div > div.a4bIc > input").send_keys('  
8         download musik mp3')  
9     driver.find_element_by_css_selector("#tsf > div:nth-child(2) >  
10        div.A8SBwf > div.FPdoLc.tfB0Bf > center > input.gNO89b").click()  
11    continue
```

method

method

```
1 if "mundur" in str(text):
2     assistant_speaks("Ok , "+ name+' membuka mundul alon-alon')
3     os.startfile("C:/Download/Compressed/Video/MUNDUR ALON ALON -
ILUX ID (OFFICIAL VIDEO) - YouTube.mp4")
4     continue
```

method

method

```
1 if "djremix" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka djremix')
3     os.startfile("C:/Download/Compressed/Video/DJ SLOW SALAH APA AKU
REMIX 2019 (VERSI GAGAK) – YouTube.mkv")
4     continue
```

method

method

```
1 if "pamer bojo" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka pamer bojo')
3     os.startfile("C:/Download/Compressed/Video/Nella Kharisma Pamer
Bojo Versi Cendol Dawet – YouTube.mkv")
4     continue
```

method

```
1 if "cendol dawet" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka cendol dawet')
3     os.startfile ("C:/Download/Compressed/Video/videoplayback.mkv")
4     continue
```

method

```
1 if "dj dimatamu" in str(text):
2     assistant_speaks("Ok , "+ name+" Membuka dj dimatamu")
3     os.startfile("C:/Download/Compressed/Video/DJ Dimatamu - Original
4     Remix Terbaru Full Bass 2019 - YouTube.mkv")
    continue
```

method

```
1 if "kopi dangdut" in str(text):
2     assistant_speaks("Ok , "+ name+" Membuka kopi dangdut")
3     os.startfile("C:\\\\Download/Compressed/Video/Fahmi Shahab - Kopi
4     Dangdut [ Official ] - YouTube.mkv")
    continue
```

method

```
1 if "korban janji" in str(text):
2     assistant_speaks("Ok , "+ name+" Membuka korban janji")
3     os.startfile("C:/Download/Compressed/Video/GuyonWaton Official -
4     Korban Janji (Official Music Video) - YouTube.mp4")
    continue
```

method

method

```
1 if "medot janji" in str(text):
2     assistant.speaks("Ok , "+ name+ ' Membuka medot janji')
3     os.startfile ("C:\Download\Compressed\Video\Lirik lagu Kartonyono
4     Medot Janji - Denny Caknan - YouTube.mkv")
    continue
```

method

method

```
1 if "sayang" in str(text):
2     assistant_speaks("Ok , "+ name+ ' Membuka sayang ')
3     os.startfile ("C:/Download/Compressed/Video/Nella Kharisma -
Sayang 2 [OFFICIAL] - YouTube.mkv")
4     continue
```

method

method

```
1 if "lintang ati" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka lintang ati')
3     os.startfile("C:/Download/Compressed/Video/Nella Kharisma
4 Pamer Bojo Versi Cendol Dawet – YouTube.mkv")
      continue
```

method

method

```
1 if "alon-alon" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka alon-alon')
3     os.startfile("C:/Download/Compressed/Video/Nella Kharisma -
4         Lintang Ati [OFFICIAL] - YouTube.mkv")
        continue
```

method

method

```
1 if "lil" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka lil nas x')
3     playsound.playsound('C:/Download/Music/Lil Nas X - Old Town
4 Road (feat. Billy Ray Cyrus) Remix.mp3')
    continue
```

method

method

```
1 if "memori" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka memori')
3     playsound.playsound('C:/Download/Music/Maroon 5 - Memories.
4     mp3')
        continue
```

method

method

```
1 if "senorita" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka senorita')
3     playsound.playsound('C:/Download/Music/Shawn Mendes , Camila
4 Cabello - Se orita.mp3')
    continue
```

method

method

```
1 if "cendol" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka cendol dawet')
3     playsound.playsound('C:/Download/Music/Nella Kharisma Pamer
4 Bojo Versi Cendol Dawet.mp3')
    continue
```

method

method

```
1 if "Lay—Lay" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka lay—lay ')
3     playsound.playsound('C:/Download/Music/DJ LAY LAY LAY ( JOKER
4 ) FULL BASS TERBARU 2019.mp3')
5     continue
```

method

method

```
1 if "Tik-Tok" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka tik-tok')
3     playsound.playsound('C:/Download/Music/DJ Tik Tok Viral 2019
4 Lagu Tik Tok Remix Terbaru 2019.mp3')
5     continue
```

method

method

```
1 if "perfect" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka perfect')
3     playsound.playsound('C:/Download/Music/Ed Sheeran - Perfect (Official Music Video).mp3')
4     continue
5 
```

method

method

```
1 if "i love you" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka i love you')
3     playsound.playsound('C:/Download/Music/Stephanie Poetri - I
4 Love You 3000 (Official Music Video).mp3')
5     continue
6 
```

method

method

```
1 if "dawet" in str(text):
2     assistant_speaks("Ok , "+ name+' Membuka cendol dawet')
3     playsound.playsound('C:/Download/Music/koplo Cendol Dawet .
4     mp3')
5     continue
```

method

method

```
1 if "keluar" in str(text) or "dadah" in str(text) or "udahan" in  
2     str(text):  
3         assistant_speaks("Ok dadah , "+ name+'.')  
4         break
```

BAB 4

PEMBANGUNAN APLIKASI

4.1 Cara penggunaan

selenium/figures/4q.png

```
File Edit Selection View Go Debug Terminal Help
croot2 (1).py - Untitled (Workspace) - Visual Studio Code
croot2 (1).py X
C:\Users\imail> cd C:\Users\imail\Downloads> & croot2 (1).py >...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell! https://aka.ms/pscore6
PS C:\Users\imail\Desktop> conda activate base
PS C:\Users\imail\Desktop> & C:\Users\imail\Anaconda\python.exe "C:/Users/imail/Downloads/croot2 (1).py"
Croot : Nama Anda Siapa ?

Python 3.7.3 64-bit [base]: conda < 3 △ 5
Ln 211, Col 26 (105 selected) Spaces: 4 | UTF-8 | CRLF | Python | ⚙
```

Gambar 4.1

selenium/figures/5q.png

A screenshot of the Visual Studio Code interface. On the left is a sidebar with icons for file operations like Open, Save, Find, and Delete. The main area is a terminal window titled 'croot2 (lipy - Untitled (Workspace))'. The terminal shows the following command-line session:

```

PS C:\Users\imail\Desktop\Valhook> cd croot2
C:\Users\imail> cd Downloads > croot2(1).py > ...
PS C:\Users\imail> cd croot2
C:\Users\imail> cd Downloads > croot2(1).py > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\imail\Desktop\Valhook> conda activate base
PS C:\Users\imail\Desktop\Valhook> & C:/Users/imail/Anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
Croot : Nama Andi Sipag ?
Bicara...

```

The status bar at the bottom indicates 'Python 3.7.3 64-bit (base):conda' and 'Ln 211, Col 26 (165 selected) Spaces: 4 -UTF-8 -CR LF Python'.

Gambar 4.2

S

selenium/figures/6q.png

A screenshot of the Visual Studio Code interface, identical to Figure 4.2. The terminal window shows the same command-line session:

```

PS C:\Users\imail\Desktop\Valhook> cd croot2
C:\Users\imail> cd Downloads > croot2(1).py > ...
PS C:\Users\imail> cd croot2
C:\Users\imail> cd Downloads > croot2(1).py > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\imail\Desktop\Valhook> conda activate base
PS C:\Users\imail\Desktop\Valhook> & C:/Users/imail/Anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
Croot : Nama Andi Sipag ?
Bicara...
Stop
Andi :
Croot : Hallo, Andi.

```

The status bar at the bottom indicates 'Python 3.7.3 64-bit (base):conda' and 'Ln 211, Col 26 (165 selected) Spaces: 4 -UTF-8 -CR LF Python'.

Gambar 4.3

S

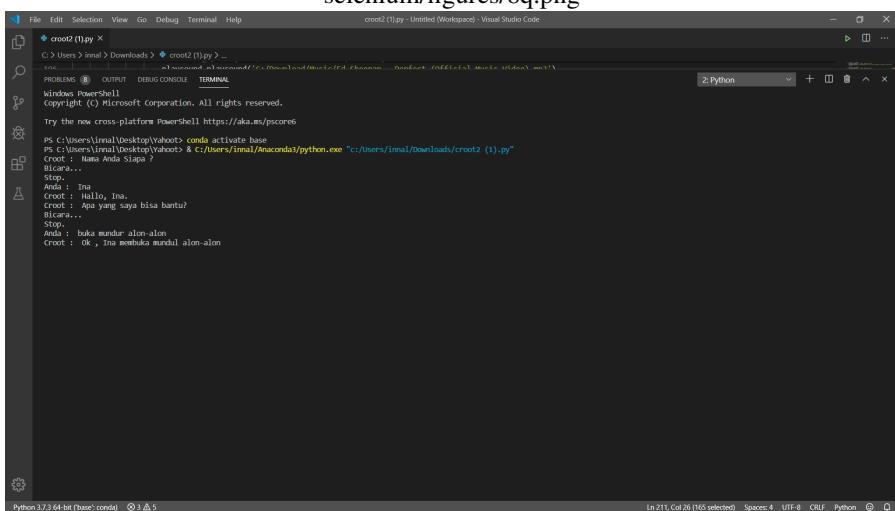
selenium/figures/7q.png

```
File Edit Selection View Go Debug Terminal Help
croot2 (1).py - Untitled (Workspace) - Visual Studio Code
croot2 (1).py > ...
C:\Users\imail>Downloads> croot2 (1).py > ...
PS C:\Users\imail> cd "C:\Users\imail\Downloads" & python croot2 (1).py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\imail\Desktop\Valohai> conda activate base
PS C:\Users\imail\Desktop\Valohai> & C:/Users/imail/Anaconda/python.exe "c:/users/imail/downloads/croot2 (1).py"
Croot : Sama Atau Sape ?
Bicara...
Anda : Ina
Croot : Hallo, Ina.
Croot : Apa yang saya bisa bantu?
Bicara...
```

Gambar 4.4

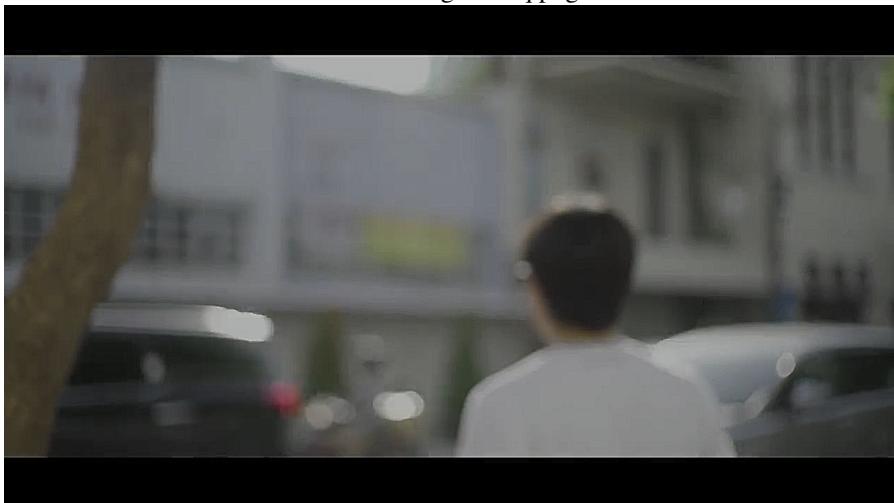
S



Gambar 4.5

S

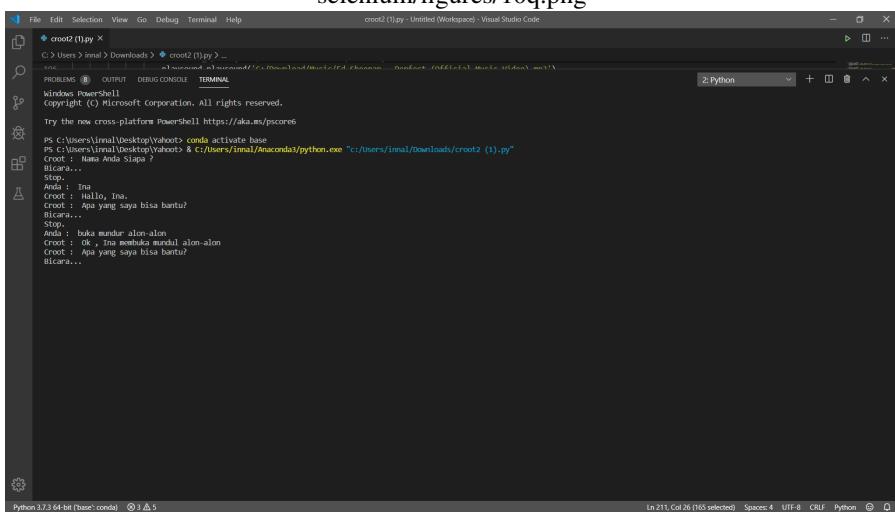
selenium/figures/9q.png



Gambar 4.6

S

S



Gambar 4.7

S

The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal title is 'croot2 (1).py'. The command entered was 'python croot2 (1).py'. The output of the script is displayed in the terminal:

```
PS C:\Users\imail\Desktop\Valent> conda activate base
PS C:\Users\imail\Desktop\Valent> & c:/users/imail/anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
Croot : Nama Anak Saya ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka mandul alom-alon
Croot : Ok , Ina membuta mandul alom-alon
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : keluar
Croot : Ok dedah, Ina.
```

Gambar 4.8

S

S

selenium/figures/12q.png

```
croot2 (1).py x
C:\Users\imail> Downloads > croot2 (1).py > ...
[snip]
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

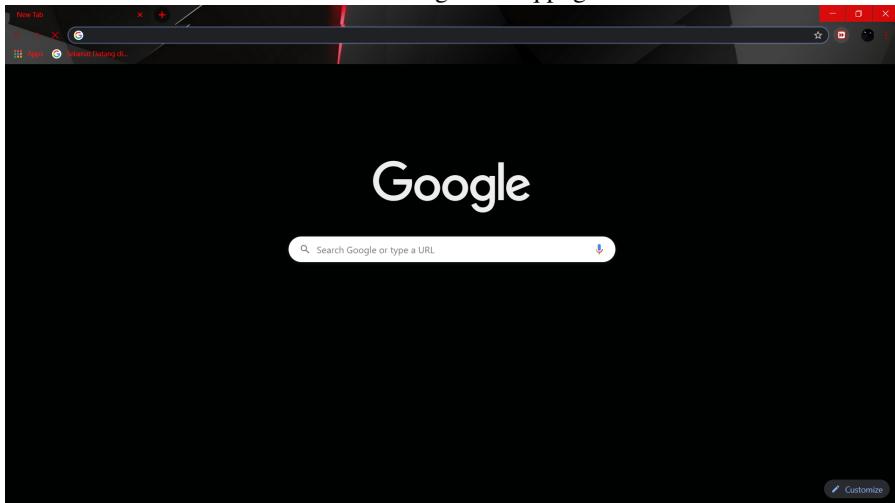
try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\imail\Desktop\Valent> conda activate base
PS C:\Users\imail\Desktop\Valent> & c:/users/imail/anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
Croot : Nama Anak Saya ?
Stop
Anda : Ina
Croot : Hallo, Ina.
Croot : Apa yang saya bisa bantu?
Inicara...
Stop
Anda : buka chrome
Croot : Ok , Ina Membuta google chrome
```

Gambar 4.9

S

S

selenium/figures/13q.png



Gambar 4.10

S

S

selenium/figures/14q.png

The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal title is "croot2 (laptop) - Untitled [Workspace] - Visual Studio Code". The terminal content shows a conversation between a user and a bot named "croot". The user types commands like "anda", "stop", and "buka chrome", and the bot responds with "hallo", "stop", and "ok". The terminal also shows the path "C:\Users\imail\Desktop\valent> & c:/users/imail/anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"" and the Python version "Python 3.7.3 64-bit (base':conda)".

```
croot2 (laptop) - Untitled [Workspace] - Visual Studio Code
PS C:\Users\imail\Desktop\valent> & c:/users/imail/anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\imail\Desktop\valent> conda activate base
PS C:\Users\imail\Desktop\valent> & c:/users/imail/anaconda3/python.exe "c:/users/imail/downloads/croot2 (1).py"
croot : Nama Ane Saja ?
Bicara...
Stop.
Anda : Ina
croot : Hallo, Ina.
croot : Apa yang saya bisa bantu?
bicara...
Stop.
Anda : buka Chrome
croot : Ok , Ina Membuka google chrome
croot : Apa yang saya bisa bantu?
Anda...
Stop.
Anda : buka spotify
croot : ok , Ina Membuka spotify

In 211 Col 0 (165 selected) Spaces: 4 -UTF-8 -CR LF Python
```

Gambar 4.11

S

S

```
croot2 (1).py x
C:\Users\imail> Downloads > croot2 (1).py > ...
PS C:\Users\imail> conda activate base
PS C:\Users\imail> cd Desktop\valent> & C:/users/imail/anaconda3/python.exe "C:/users/imail/Downloads/croot2 (1).py"
[REMOVED] ① OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\imail\Desktop\valent> conda activate base
PS C:\Users\imail> cd Desktop\valent> & C:/users/imail/anaconda3/python.exe "C:/users/imail/Downloads/croot2 (1).py"
Croot : Nama Anu Sipagi ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Anda : .
Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify

Ln 211, Col 0 (165 selected) Spaces: 4 -UTF-8 -CR LF Python
```

Gambar 4.12

S

S

selenium/figures/16q.png

The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal title is 'croot2 (1)'. The command entered was 'python croot2 (1).py'. The output of the script is displayed in the terminal:

```
C:\Users\Imai> python croot2 (1).py
...
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\Imai\Desktop\Valent> conda activate base
PS C:\Users\Imai\Desktop\Valent> & c:/users/imai/anaconda3/python.exe "c:/users/imai/downloads/croot2 (1).py"
...
Anda : Nama Ani Sipage ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka youtube
Croot : Apa yang saya bisa bantu?
```

The terminal also shows the status bar at the bottom indicating 'Ln 211 Col 0 (165 selected) Spaces: 4 -UTF-8 -CR LF Python'.

Gambar 4.13

S

S

selenium/figures/17.png

```

File Edit Selection View Go Debug Terminal Help
croot2 (1).py x
C:\Users\imail> Downloads > croot2 (1).py > ...
Windows PowerShell [D:\croot2]
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\imail\Desktop\Valent> conda activate base
PS C:\Users\imail\Desktop\Valent> & C:/users/imail/anaconda3/python.exe "C:/users/imail/downloads/croot2 (1).py"
Croot : Nama Anak Saya ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka youtube
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka musik
Croot : Ok , Ina Membuka mundur alon-alon

```

Python 3.7.3 64-bit ('base':conda) ⚡ 3 △ 5

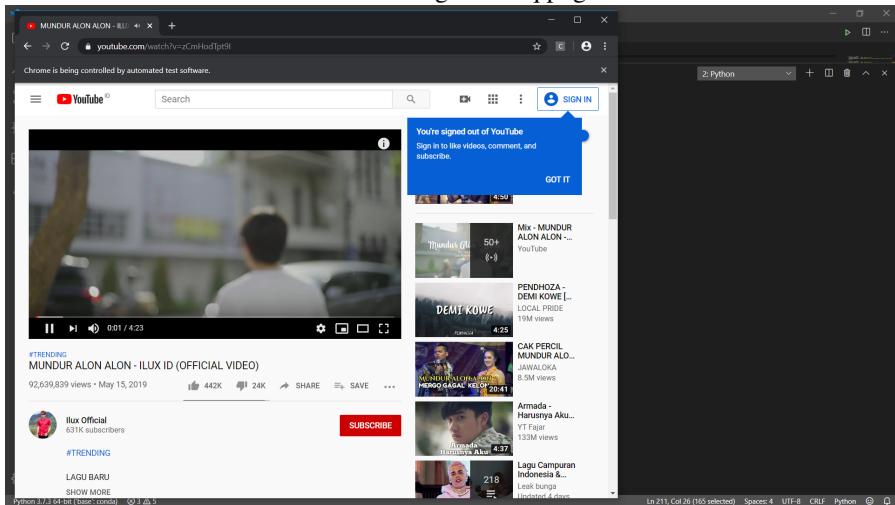
In 211, Col 0 (165 selected) Spaces: 4 -UTF-8 -CR LF Python ⚡ Q

Gambar 4.14

S

S

selenium/figures/18q.png



Gambar 4.15

S

S

selenium/figures/19q.png

```

File Edit Selection View Go Debug Terminal Help
croot2 (1).py x
C:\Users\imail> Downloads > croot2 (1).py > ...
PS C:\Users\imail> conda activate base
PS C:\Users\imail> cd Desktop\valent< & C:\Users\imail\Anaconda3\python.exe "C:/Users/imail/Downloads/croot2 (1).py"
Croot : Nama Anak Saya ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka youtube
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka musik
Croot : Ok , Ina Membuka mundur alon-alon
c:/users/imail/downloads/croot2 (1).py:59: DeprecationWarning: use options instead of chrome_options
      driver = webdriver.Chrome(chrome_options=options)

DevTools listening on ws://127.0.0.1:6563/devtools/browser/817e0bec-a223-4259-99e1-3af7a7a0f918
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : keluar
Croot : Ok dadah, Ina.

Python 3.7.3 64-bit ('base':conda) ① 3 △ 5

```

In 211, Col 0 (165 selected) Spaces: 4 -UTF-8 -CR LF Python ① Q

Gambar 4.16

S

S S

S

selenium/figures/20q.png

tutorial selenium/figures/20q.png

Gambar 4.17

S

S

selenium/figures/19q.png

```
croot2 (1).py
C:\Users\Inna> Downloads > croot2 (1).py > ...
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Inna> conda activate base
PS C:\Users\Inna> & C:\Users\Inna\Downloads\python.exe "C:/Users/Inna/Downloads/croot2 (1).py"
Croot : Nama Anda Saja ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka youtube
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka musik
Croot : Ok , Ina Membuka mundur alon-alon
c:/users/inna/downloads/croot2 (1).py:59: DeprecationWarning: use options instead of chrome_options
      driver = webdriver.Chrome(chrome_options=options)

DevTools listening on ws://127.0.0.1:6563/devtools/browser/817e0bec-a223-4259-99e1-3af7a/a0f918
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : keluar
Croot : Ok dadah, Ina.
```

Gambar 4.18

S

selenium/figures/19q.png

The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal title is 'croot2 (1)py'. The terminal content displays a conversation between two entities, 'Anda' and 'Croot'. 'Anda' asks various questions such as 'Buka browser?', 'Buka youtube?', 'Buka spotify?', and 'Buka musik?'. 'Croot' responds to each question with 'Ok, Ina'. The terminal also shows command-line navigation and some system logs at the bottom.

```
C:\> cd C:\Users\Inal> Downloads > croot2 (1).py > ...
PS C:\Users\Inal> cd C:\Users\Inal> & c:/users/inal/Downloads/cpython.exe "c:/users/inal/downloads/croot2 (1).py"
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Inal> conda activate base
PS C:\Users\Inal> cd C:\Users\Inal> & c:/users/inal/Downloads/cpython.exe "c:/users/inal/downloads/croot2 (1).py"
Croot : Nama Ane Saja ?
Bicara...
Stop.
Anda : Ina
Croot : Hallo, Ina.
Anda : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka Chrome
Croot : Ok , Ina Membuka google chrome
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka spotify
Croot : Ok , Ina Membuka spotify
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : buka youtube
Croot : Apa yang saya bisa bantu?
Anda : Stop.
Anda : buka musik
Croot : Ok , Ina Membuka mundur alon-alon
c:/users/inal/Downloads/croot2 (1).py:59: DeprecationWarning: use options instead of chrome_options
      driver = webdriver.Chrome(chrome_options=options)

DevTools listening on ws://127.0.0.1:6563/devtools/browser/817e0dec-a223-4259-99e1-3af7a/a0f918
Croot : Apa yang saya bisa bantu?
Bicara...
Stop.
Anda : keluar
Croot : Ok dadah, Ina.
```

Gambar 4.19

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.

