

Python Programming

Loop and Iteration



Topik *Python Programming*

Topik 1 : *Conditional Statement*

Topik 2 : *Loop and Iteration*

Topik 3 : *Function and Error Handling*

Topik Sebelumnya : **Conditional Statement**



Buatlah **syntax If** untuk mengecek apakah plat nomor **B 1256 AE** masuk plat genap atau ganjil !



Topik Sebelumnya : Conditional Statement



Buatlah **syntax If** untuk mengecek apakah plat nomor **B 1256 AE** masuk plat genap atau ganjil !

Bagaimana jika kita ingin melakukan pengecekan plat nomor **seluruh warga Jakarta** sekaligus?



Loop and Iteration

- ☐ Pengenalan Iterasi
- ☐ Definite Iteration
- ☐ Indefinite Iteration
- ☐ Pembahasan Mini Quiz Iterasi #1
- ☐ Iterasi pada List - Definite Iteration
- ☐ One liner dan Nested Definite Iteration
- ☐ Iterasi pada List - Indefinite Iteration
- ☐ Pembahasan Mini Quiz Iterasi #2
- ☐ Iterasi pada Dictionary
- ☐ Pembahasan Mini Quiz Iterasi #3
- ☐ Iterasi pada Dataframe
- ☐ Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Hands-On Requirement:

Hands - On :

Python Programming 2 - Rakamin Academy.ipynb

Dataset :

monthly_rakamin_customer_order.csv

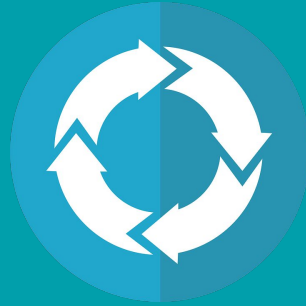
rakamin_customer_week1.csv

rakamin_customer_week2.csv

rakamin_customer_week3.csv

**Klik disini untuk mengakses
folder Hands-On dan
Dataset**

Pengenalan *Iterasi*



*“Iterasi berarti **mengeksekusi blok kode yang sama berulang-ulang sejumlah n kali (n kali possible infinite iterasi)**. Dimana sebuah kode pemrograman yang mengimplementasikan iterasi disebut **loop**”*

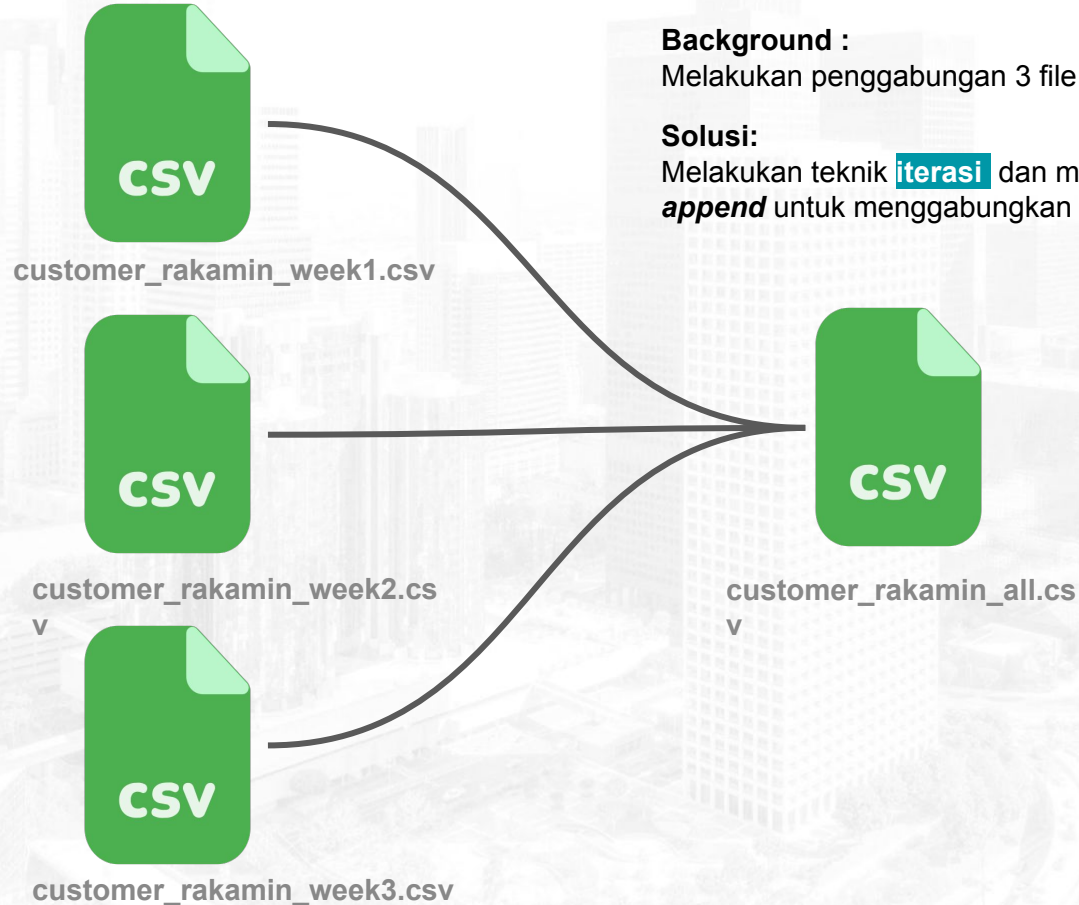
Salah satu contoh penerapan materi iterasi

Background :

Melakukan penggabungan 3 file csv atau lebih menjadi 1 file csv

Solusi:

Melakukan teknik **iterasi** dan menggunakan fungsi **append** untuk menggabungkan 3 file csv



Mengapa Iterasi Penting?

Bayangkan! jika kita memiliki 1 juta item list yang perlu dilakukan perulangan? Apakah kita akan menggunakan cara pada **gambar kiri?**

```
1 list_angka = [1,2,3,4,5,6,7,8,9]
2
3 print(list_angka[0])
4 print(list_angka[1])
5 print(list_angka[2])
6 print(list_angka[3])
7 print(list_angka[4])
8 print(list_angka[5])
9 print(list_angka[6])
10 print(list_angka[7])
11 print(list_angka[8])
```

1
2
3
4
5
6
7
8
9



```
1 list_angka = [1,2,3,4,5,6,7,8,9]
2
3 for i in list_angka:
4     print(i)
```

1
2
3
4
5
6
7
8
9

More effective!

Cukup dengan 2
baris kode *for loop*

Terdapat 2 Macam Perulangan



```
graph TD; A[Terdapat 2 Macam Perulangan] --> B[Definite Iteration]; A --> C[Indefinite Iteration];
```

Definite Iteration

Indefinite Iteration

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Terdapat 2 Macam Perulangan



```
graph TD; A[Terdapat 2 Macam Perulangan] --> B[Definite Iteration]; A --> C[Indefinite Iteration];
```

Definite Iteration

Indefinite Iteration

Terdapat 2 Macam Perulangan



```
graph TD; A[Terdapat 2 Macam Perulangan] --> B[Definite Iteration]; A --> C[Indefinite Iteration];
```

Definite Iteration

Indefinite Iteration

Definite Iteration

Definite Iteration adalah perulangan yang **jumlah iterasinya ditentukan secara eksplisit** di awal. Contoh **definite iteration** adalah **for loop**

Contoh Code:

```
1 for target in iterable:
2     statement(s)
```

Implementasi source code for loop

- Header line (line 1) pada for loop diawali dengan **syntax for** diikuti dengan **object target** lalu **syntax in** dan **suatu object iterable** diakhiri dengan **titik dua** :

Definite Iteration

Definite Iteration adalah perulangan yang **jumlah iterasinya ditentukan secara eksplisit** di awal. Contoh **definite iteration** adalah **for loop**

Contoh Code:

```
1 for target in iterable:
2     statement(s)
```

Implementasi source code for loop

- Header line (line 1) pada for loop diawali dengan **syntax for** diikuti dengan **object target** lalu **syntax in** dan **suatu object iterable** diakhiri dengan **titik dua** :
- **Object iterable** adalah object yang dapat melakukan proses iterasi

Definite Iteration

Definite Iteration adalah perulangan yang **jumlah iterasinya ditentukan secara eksplisit** di awal. Contoh **definite iteration** adalah **for loop**

Contoh Code:

```
1 for target in iterable:
2     statement(s)
```

Implementasi source code for loop

- Header line (line 1) pada for loop diawali dengan **syntax for** diikuti dengan **object target** lalu **syntax in** dan **suatu object iterable** diakhiri dengan **titik dua** :
- **Object iterable** adalah object yang dapat melakukan proses iterasi
- **Body For loop** bisa terdiri dari 1 atau lebih **statements**

Definite Iteration

```
1 for target in iterable:  
2     statement(s)
```

Definite Iteration melakukan perulangan pada **body loop sejumlah item pada object iterable**. Nilai **target** bernilai item pada object iterable setiap iterasinya.

Definite Iteration

Contoh 1

```
1 for target in iterable:
2     statement(s)
```

```
1 list_angka = [1,2,3,4,5]
2
3 for i in list_angka:
4     print(i)
```

```
1
2
3
4
5
```

Definite Iteration melakukan perulangan pada **body loop sejumlah item pada object iterable**. Nilai **target** bernilai item pada object iterable setiap iterasinya.

Definite Iteration

Contoh 1

```
1 list_angka = [1,2,3,4,5]
2
3 for i in list_angka:
4     print(i)
```

```
1
2
3
4
5
```

```
1 for target in iterable:
2     statement(s)
```

Contoh 2

```
1 name = 'rakamin'
2
3 for m in name:
4     print(m)
```

```
r
a
k
a
m
i
n
```

Definite Iteration melakukan perulangan pada **body loop sejumlah item pada object iterable**. Nilai **target** bernilai item pada object iterable setiap iterasinya.

Iterable

Dalam Bahasa Pemrograman Python, *Iterable* adalah suatu object yang dapat digunakan untuk iterasi

Kita dapat mengetahui suatu object termasuk dalam kategori iterable jika berhasil lolos dalam pengecekan menggunakan fungsi [iter\(\)](#).

```

Python

>>> iter('foobar')                                # String
<str_iterator object at 0x036E2750>

>>> iter(['foo', 'bar', 'baz'])                    # List
<list_iterator object at 0x036E27D0>

>>> iter(('foo', 'bar', 'baz'))                    # Tuple
<tuple_iterator object at 0x036E27F0>

>>> iter({'foo', 'bar', 'baz'})                     # Set
<set_iterator object at 0x036DEA08>

>>> iter({'foo': 1, 'bar': 2, 'baz': 3})            # Dict
<dict_keyiterator object at 0x036DD990>

```

[Reference](#)

For Loop - Menggunakan `.range()`

Pada beberapa kasus kita membutuhkan kondisi perulangan yang jumlahnya **pasti dan konsisten**. Pada kasusnya, kita bisa dimudahkan dengan fungsi *built-in* pada Python, yaitu **`.range()`**

Contoh kasus:

Kondisi	Source Code	Notes
Melakukan perulangan sejumlah 100 kali	<pre>for i in range(100): print(i)</pre> <pre>for i in range(1, 101): print(i)</pre>	<p>Contoh 1 parameter: akan menghasilkan angka dari 0 sampai 99. Jika dijumlahkan melakukan 100 kali iterasi</p> <p>Contoh 2 parameter: akan melakukan iterasi dari 1 sampai 100, karena <code>range(a, b)</code> menghasilkan dari deret angka dari a sampai b-1. Jika dijumlahkan melakukan 100 kali iterasi</p>
Melakukan perulangan start dari 2 sampai 100 setiap kelipatan 2	<pre>for i in range(2, 101, 2): print(i)</pre>	Akan melakukan iterasi dengan dengan nilai i dimulai dengan 2 kemudian 2+2 (4) hingga 101-1, karena <code>range(a, b, step=n)</code> membuat adanya kondisi kelipatan 2.

For Loop - Menggunakan `.range()`

Snippet Code

Contoh 1

```
1 for i in range(1, 101):
2     print(i)
```

1
2
3
4
5
6
7

Contoh 2

```
1 for i in range(2, 101, 2):
2     print(i)
```

2
4
6
8
10
12
14
16

Secara concept fungsi `.range()` ini kita seperti membuat sebuah **list** dengan nilai item adalah **deret angka yang berurut**.

For Loop - Lebih dari 1 Statement Contoh 1

Jumlah **statement** pada perulangan **for** bisa lebih dari 1.

Contoh kasus : Membuat bilangan kuadrat dari 1-10 dan mencetak hasilnya

```
1 for i in range(1, 11):
2     kuadrat = i*i #statement 1
3     print('Kuadrat', i, 'adalah', kuadrat) #statement 2
```

For Loop - Lebih dari 1 Statement Contoh 1

Jumlah **statement** pada perulangan **for** bisa lebih dari 1.

Contoh kasus : Membuat bilangan kuadrat dari 1-10 dan mencetak hasilnya

```
1 for i in range(1, 11):
2     kuadrat = i*i #statement 1
3     print('Kuadrat', i, 'adalah', kuadrat) #statement 2
```

```
Kuadrat 1 adalah 1
Kuadrat 2 adalah 4
Kuadrat 3 adalah 9
Kuadrat 4 adalah 16
Kuadrat 5 adalah 25
Kuadrat 6 adalah 36
Kuadrat 7 adalah 49
Kuadrat 8 adalah 64
Kuadrat 9 adalah 81
Kuadrat 10 adalah 100
```

For Loop - Lebih dari 1 Statement Contoh 2

Jumlah **statement** pada perulangan **for** bisa lebih dari 1.

Contoh kasus : Membuat nama kolom menjadi huruf kecil (*lower case*) dan *snake case* (_)

df		
	Nama Customer	Kota Customer
0	Ammar	Surabaya
1	Hasan	Bandung



df		
	nama_customer	kota_customer
0	Ammar	Surabaya
1	Hasan	Bandung

For Loop - Lebih dari 1 Statement Contoh 2

Jumlah **statement** pada perulangan **for** bisa lebih dari 1.

Contoh kasus : Membuat nama kolom menjadi huruf kecil (*lower case*) dan *snake case* (*_*)

```

1 column_name = df.columns
2 column_name

Index(['Nama Customer', 'Kota Customer'], dtype='object')

1 column_name_new = [] #list_kosong
2
3 # iterasi setiap nama column
4 for i in column_name:
5     new_name = i.lower() #statement 1
6     split_name = new_name.split() #[name1, name2] #statement 2
7     new_name = split_name[0] + '_' + split_name[1] #statement 3
8     column_name_new.append(new_name) #statement 4
9
10 print(column_name_new)

```

For Loop - Lebih dari 1 Statement Contoh 2

Jumlah **statement** pada perulangan **for** bisa lebih dari 1.

Contoh kasus : Membuat nama kolom menjadi huruf kecil (*lower case*) dan *snake case* (*_*)

```

1 column_name = df.columns
2 column_name

Index(['Nama Customer', 'Kota Customer'], dtype='object')

1 column_name_new = [] #list_kosong
2
3 # iterasi setiap nama column
4 for i in column_name:
5     new_name = i.lower() #statement 1
6     split_name = new_name.split() #[name1, name2] #statement 2
7     new_name = split_name[0] + '_' + split_name[1] #statement 3
8     column_name_new.append(new_name) #statement 4
9
10 print(column_name_new)

['nama_customer', 'kota_customer']

```

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Terdapat 2 Macam Perulangan



```
graph TD; A[Terdapat 2 Macam Perulangan] --> B[Definite Iteration]; A --> C[Indefinite Iteration];
```

Definite Iteration

Indefinite Iteration

Indefinite Iteration

Indefinite Iteration adalah perulangan yang terus berulang **sampai tidak memenuhi kondisi** tertentu (False). Contoh **indefinite iteration** adalah **while loops**

Contoh Code:

```
1 while expression:  
2     statement(s)
```

Implementasi source code *while loop*

- Header line (line 1) pada while loop diawali dengan **syntax while** diikuti dengan suatu **expression** yang umumnya bertipe *boolean* dan diakhiri dengan *titik dua* :

Indefinite Iteration

Indefinite Iteration adalah perulangan yang terus berulang **sampai tidak memenuhi kondisi** tertentu (False). Contoh **indefinite iteration** adalah **while loops**

Contoh Code:

```
1 while expression:
2     statement(s)
```

Implementasi source code *while loop*

- Header line (line 1) pada while loop diawali dengan **syntax while** diikuti dengan suatu **expression** yang umumnya bertipe *boolean* dan diakhiri dengan *titik dua* :
- **Body While loop** bisa terdiri dari 1 atau lebih *statements*

Indefinite Iteration

```
1 while expression:  
2     statement(s)
```

Indefinite Iteration melakukan perulangan hingga expression bernilai **False**. Sehingga pada pembuatan *while loop* **perlu ada logic yang membuat expression memiliki kemungkinan untuk berhenti (ex. $i+=1$)**.

Indefinite Iteration

Contoh 1

```
1 while expression:
2     statement(s)
```

```
1 #while loop
2 i = 0 #initiate var
3 while i < 7: #expression
4     print(i) #statement 1
5     i+=1 #statement 2 | i=i+1
```

```
0
1
2
3
4
5
6
```

Indefinite Iteration melakukan perulangan hingga expression bernilai **False**. Sehingga pada pembuatan *while loop* **perlu ada logic yang membuat expression memiliki kemungkinan untuk berhenti (ex. i+=1)**.

Indefinite Iteration

```
1 while expression:
2     statement(s)
```

Contoh 1

```
1 #while loop
2 i = 0 #initiate var
3 while i < 7: #expression
4     print(i) #statement 1
5     i+=1 #statement 2 | i=i+1
```

```
0
1
2
3
4
5
6
```

Contoh 2

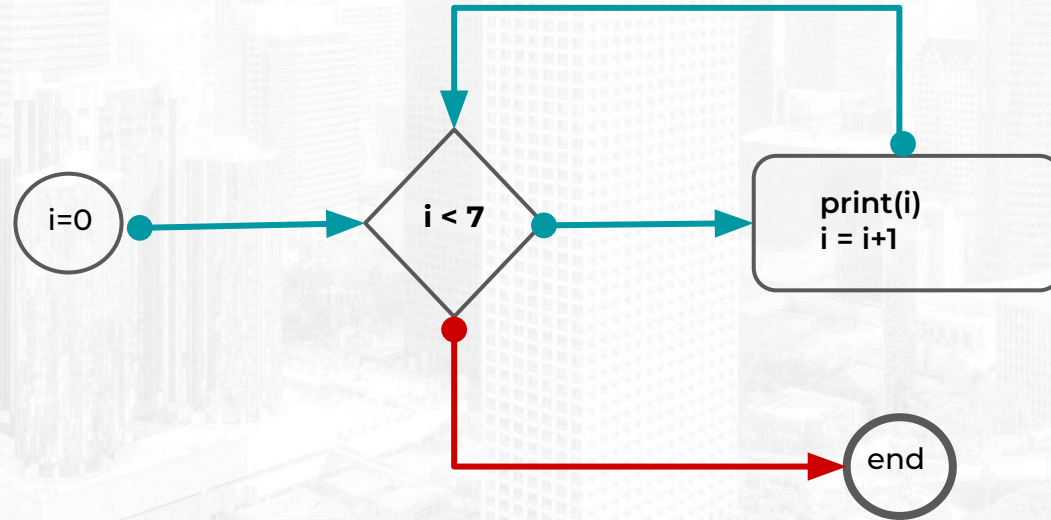
```
1 #while loop
2 list_number = range(7)
3 i = 0 #initiate var
4 while i < len(list_number): #expression
5     print(i) #statement 1
6     i+=1 #statement 2 | i=i+1
```

```
0
1
2
3
4
5
6
```

Indefinite Iteration melakukan perulangan hingga expression bernilai **False**. Sehingga pada pembuatan *while loop* **perlu ada logic yang membuat expression memiliki kemungkinan untuk berhenti (ex. i+=1)**.

Flow Indefinite Iteration

while loops diagram



Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Terdapat 50 user dengan **user_id berurut dari 101 - 150**, kita akan memilih beberapa user_id saja yang akan dijadikan sebagai target user interview. Kriteria user yang akan diambil adalah.

Jika nilai user_id adalah bilangan kelipatan 5 yang habis dibagi 2.

Ada berapa jumlah user yang menjadi target participant? Buatlah dua versi, pertama menggunakan iterasi for dan kedua menggunakan while!

Hint: Buatlah variabel dengan value 0 yang terus bertambah secara incremental setiap iterasi jika memenuhi kondisi yang diminta, gunakan ***conditional statement - IF***



Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3

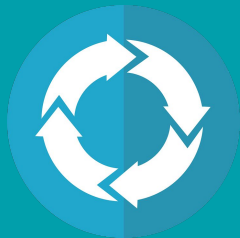


Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Iterasi pada **List**



*“Iterasi pada **list** berfungsi untuk melakukan perulangan terhadap **setiap itemnya**”*

Iterasi pada **Type Data** **List**



```
graph TD; A[Iterasi pada Type Data List] --- B[For Loop]; A --- C[While Loop];
```

For Loop

While Loop

Iterasi pada **Type Data List**



```
graph TD; A[Iterasi pada Type Data List] --- B[For Loop]; A --- C[While Loop];
```

For Loop

While Loop

For Loop pada List

Contoh kasus: Ketika kita ingin mengecek apakah setiap item pada **list_nilai** lebih besar dari standar kelulusan 70

Contoh Code:

```
1 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
2
3 for i in list_nilai:
4     cek_nilai = i > 70 #melakukan pengecekan boolean
5     print(cek_nilai)
```

```
False
True
False
True
True
True
True
True
```

Notes

- Nilai **i** akan **bernilai setiap item** pada variabel **list_nilai** setiap iterasi (perulangannya), dimulai dari index ke 0, 1, 2 dan seterusnya.

For Loop pada List

Alur pengecekan yang terjadi sebagai berikut:

```

1 list_nilai = [75, 76, 83, 66, 68]
2
3 for i in list_nilai:
4     cek_nilai = i > 70
5     print(cek_nilai)

```

True
True
True
False
False



```

# Alur pengecekan
cek_nilai = list_nilai[0]>70
print(cek_nilai)

cek_nilai = list_nilai[1]>70
print(cek_nilai)

cek_nilai = list_nilai[2]>70
print(cek_nilai)

cek_nilai = list_nilai[3]>70
print(cek_nilai)

cek_nilai = list_nilai[4]>70
print(cek_nilai)

```

Memunculkan Index pada Iterasi List - `.index()`

Pada beberapa kasus selain nilai pada item list kita juga membutuhkan untuk mendapatkan nilai indexnya. Salah satu cara untuk mendapatkan nilai index adalah dengan fungsi *built-in* `.index()`

Contoh Code:

```
1 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
2
3 for i in list_nilai:
4     print('index', list_nilai.index(i), 'adalah', i)
```

```
index 0 adalah 70
index 1 adalah 72
index 2 adalah 60
index 3 adalah 80
index 4 adalah 83
index 5 adalah 90
index 6 adalah 85
index 7 adalah 100
```

Notes

- `i` dalam for loop akan bernilai item pada setiap perulangannya
- untuk memunculkan index kita bisa menggunakan fungsi `.index()`

Memunculkan Index pada Iterasi List - `.index()`

Pada beberapa kasus selain nilai pada item list kita juga membutuhkan untuk mendapatkan nilai indexnya. Salah satu cara untuk mendapatkan nilai index adalah dengan fungsi *built-in* `.index()`

Contoh Code:

```
1 #memunculkan index dengan fungsi .index()
2 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100, 72]
3
4 for i in list_nilai:
5     print('index', list_nilai.index(i), 'adalah', i)
```

```
index 0 adalah 70
index 1 adalah 72
index 2 adalah 60
index 3 adalah 80
index 4 adalah 83
index 5 adalah 90
index 6 adalah 85
index 7 adalah 100
index 1 adalah 72
```

Notes

- `i` dalam for loop akan bernilai item pada setiap perulangannya
- untuk memunculkan index kita bisa menggunakan fungsi `.index()`
- **Namun kekurangannya** jika dalam kondisi beberapa index memiliki value yang sama

Memunculkan Index pada Iterasi List - `enumerate()`

Pada beberapa kasus selain nilai pada item kita juga membutuhkan untuk mendapatkan nilai indexnya. Selain `.index()` salah satu cara yang bisa kita gunakan adalah fungsi *built-in* `enumerate()`

Contoh Code:

```
1 #memunculkan index dengan fungsi .enumerate()
2 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100, 72]
3
4 for i, x in enumerate(list_nilai):
5     print('index', i, 'adalah', x)
```

```
index 0 adalah 70
index 1 adalah 72
index 2 adalah 60
index 3 adalah 80
index 4 adalah 83
index 5 adalah 90
index 6 adalah 85
index 7 adalah 100
index 8 adalah 72
```

Notes

- **Target memiliki 2 variabel (`i` dan `x`)**, dimana `i` adalah index dan `x` bernilai item pada setiap index
- untuk memunculkan index kita menggunakan fungsi **`enumerate()`**

Membuat List Baru pada for Loop

Pada beberapa kasus kita akan butuh hasil list baru dari suatu iterasi.
Contoh Kasus : Membuat **list baru** yang bernilai status hasil ujian setiap siswa (Lulus : Jika nilai > 75)

Contoh Code:

```
1 #Studi kasus: membuat list baru yang merupakan status hasil ujian
2
3 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
4 status = [] #list kosong
5
6 for i in list_nilai:
7     if i > 75:
8         status.append('Lulus')
9     else:
10        status.append('Tidak Lulus')
11
12 status
```

```
['Tidak Lulus',
'Tidak Lulus',
'Tidak Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus']
```

Notes (Logic)

- Membuat **list kosong** untuk menampung item baru (Lulus / Tidak Lulus) setiap iterasi
- Menggunakan fungsi **.append** untuk menambahkan item baru pada variabel **status**

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Kira-kira bisa **lebih dipersingkat** lagi gak ya syntax nya?

```
1 #Studi kasus: membuat list baru yang merupakan status hasil ujian
2
3 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
4 status = [] #list kosong
5
6 for i in list_nilai:
7     if i > 75:
8         status.append('Lulus')
9     else:
10        status.append('Tidak Lulus')
11
12 status
```

```
['Tidak lulus',
'Tidak lulus',
'Tidak lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus']
```



One Liner | List Comprehension

Pada beberapa kasus kita bisa meng efisiensikan baris kode yang kita buat untuk menghasilkan suatu list baru pada suatu iterasi dengan concept *list comprehension*. Hal ini membuat **for loop pada list menjadi lebih singkat**

One Liner | List Comprehension

Pada beberapa kasus kita bisa meng efisiensikan baris kode yang kita buat untuk menghasilkan suatu list baru pada suatu iterasi dengan concept *list comprehension*. Hal ini membuat **for loop pada list menjadi lebih singkat**

Perbandingan Code:

```

1  #Studi kasus: membuat list baru yang merupakan status hasil ujian
2
3  list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
4  status = [] #list kosong
5
6  for i in list_nilai:
7      if i > 75:
8          status.append('Lulus')
9      else:
10         status.append('Tidak Lulus')
11
12  status

['Tidak Lulus',
'Tidak Lulus',
'Tidak Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus']

```



```

1  #One liner | list comprehension
2  list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
3
4  status = ['Lulus' if i > 75 else 'Tidak Lulus' for i in list_nilai]
5  status

['Tidak Lulus',
'Tidak Lulus',
'Tidak Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus']

```


One Liner | List Comprehension

Pada beberapa kasus kita bisa meng efisiensikan baris kode yang kita buat untuk menghasilkan suatu list baru pada suatu iterasi dengan concept *list comprehension*. Hal ini membuat **for loop pada list menjadi lebih singkat**

Code:

```
1 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
2
3 #list comprehension | one liner
4 # status = [<Conditional statement one liner> for target in iterables]
```

One Liner | List Comprehension

Conditional statement setiap item iterasi

```
1 #One liner / list comprehension
2 list_nilai = [70, 72, 60, 80, 83, 90, 85, 100]
3
4 status = ['Lulus' if i > 75 else 'Tidak Lulus' for i in list_nilai]
5 status
```

```
['Tidak Lulus',
'Tidak Lulus',
'Tidak Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus',
'Lulus']
```

Syntax iterasi one liner (**tanpa :**)

Notes: List Comprehension ini diawali dengan **kurung siku []**

Nested For - Pada Context List

Pada beberapa kondisi kita membutuhkan case **for di dalam for (nested for)**. **Contoh kasus:** Pasangan bundling category, dimana bundling wajib dengan kategori yang berbeda.

Ilustrasi Bundling

Price = 49\$

Price = 99\$

Total Price = ~~148\$~~
110\$

Nested For - Pada Context List

Pada beberapa kondisi kita membutuhkan case for di dalam for (**nested for**).

Contoh kasus: Pasangan bundling category, dimana bundling wajib dengan kategori yang berbeda.

Contoh Code:

```
1 list_category = ['Fashion', 'Makanan', 'Handphone', 'Sepeda', 'Rumah Tangga']  
2 bundling_list = []
```

Nested For - Pada Context List

Pada beberapa kondisi kita membutuhkan case for di dalam for (**nested for**).

Contoh kasus: Pasangan bundling category, dimana bundling wajib dengan kategori yang berbeda.

```

1 list_category = ['Fashion', 'Makanan', 'Handphone', 'Sepeda', 'Rumah Tangga']
2 bundling_list = []
3
4 for i in list_category:
5     category1 = i #category pertama
6     for y in list_category:
7         if i != y: #pengecekan beda category
8             category2 = y #category kedua
9             bundling = i + '-' + y
10            bundling_list.append(bundling)
11
12 bundling_list

```

Notes

- **for pertama** untuk mendapatkan kategori pertama
- **for kedua** untuk mendapatkan pasangan kategori kedua

Nested For - Pada Context List

Snippet Hasil Source Code

```

1 list_category = ['Fashion', 'Makanan', 'Handphone', 'Sepeda', 'Rumah Tangga']
2 bundling_list = []
3
4 for i in list_category:
5     category1 = i #category pertama
6     for y in list_category:
7         if i != y: #pengecekan beda category
8             category2 = y #category kedua
9             bundling = i + '-' + y
10            bundling_list.append(bundling)

```



12 bundling_list

```

['Fashion-Makanan',
 'Fashion-Handphone',
 'Fashion-Sepeda',
 'Fashion-Rumah Tangga',
 'Makanan-Fashion',
 'Makanan-Handphone',
 'Makanan-Sepeda',
 'Makanan-Rumah Tangga',
 'Handphone-Fashion',
 'Handphone-Makanan',
 'Handphone-Sepeda',
 'Handphone-Rumah Tangga',
 'Sepeda-Fashion',
 'Sepeda-Makanan',
 'Sepeda-Handphone',
 'Sepeda-Rumah Tangga',
 'Rumah Tangga-Fashion',
 'Rumah Tangga-Makanan',
 'Rumah Tangga-Handphone',
 'Rumah Tangga-Sepeda']

```


Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Iterasi pada **Type Data List**



While Loop pada List

Untuk melakukan iterasi sejumlah item list pada while loop, kita bisa **menggunakan fungsi len()** yang dijadikan pada kondisi **expression**.

Contoh Code:

```
1 list_angka = [1,2,3,4,5,6,7]
2
3 i = 0
4 while i < len(list_angka):
5     print(list_angka[i])
6     i += 1 # i=i+1
```

```
1
2
3
4
5
6
7
```

While Loop pada List

Untuk melakukan iterasi sejumlah item list pada while loop, kita bisa **menggunakan fungsi len()** yang dijadikan pada kondisi **expression**.

Contoh Code:

```
1 list_angka = [1,2,3,4,5,6,7]
2
3 i = 0
4 while i < len(list_angka):
5     print(list_angka[i])
6     i += 1 # i=i+1
```

1
2
3
4
5
6
7

Notes

Saat pembuatan *while loop* perlu ada logic yang membuat expression **memiliki kemungkinan untuk berhenti** (ex. $i+=1$).

While Loop pada List

Untuk melakukan iterasi sejumlah item list pada while loop, kita bisa **menggunakan fungsi len()** yang dijadikan pada kondisi **expression**.

Contoh Code:

1

```
1 list_angka = [1,2,3,4,5,6,7]
2
3 i = 0
4 while i < len(list_angka):
5     print(list_angka[i])
6     i += 1 # i=i+1
```

1
2
3
4
5
6
7

2

```
1 list_category = ['Fashion', 'Buku', 'Makanan', 'Sepeda']
2
3 i = 0
4 while i < len(list_category):
5     print(list_category[i])
6     i += 1 # i=i+1
```

Fashion
Buku
Makanan
Sepeda

While Loop pada List

Untuk melakukan iterasi sejumlah item list pada while loop, kita bisa **menggunakan fungsi len()** yang dijadikan pada kondisi **expression**.

Contoh Code:

1

```

1 list_angka = [1,2,3,4,5,6,7]
2
3 i = 0
4 while i < len(list_angka):
5     print(list_angka[i])
6     i += 1 # i=i+1

```

1
2
3
4
5
6
7

2

```

1 list_category = ['Fashion', 'Buku', 'Makanan', 'Sepeda']
2
3 i = 0
4 while i < len(list_category):
5     print(list_category[i])
6     i += 1 # i=i+1

```

Fashion
Buku
Makanan
Sepeda

Notes

Saat pembuatan *while loop* perlu ada logic yang membuat expression **memiliki kemungkinan untuk berhenti** (ex. `i+=1`).

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Berikut adalah list **total order customer Rakamin Store**

list_total_order = [10, 2, 18, 30, 56, 12, 8, 20, 45, 60, 10, 23, 3]

Tentukan ada berapa banyak customer yang **memiliki nilai order diatas rata-rata customer Rakamin Store!** Gunakan cara for atau while



Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3

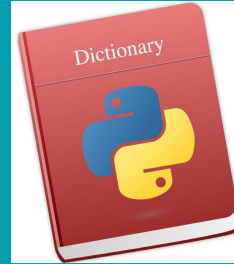
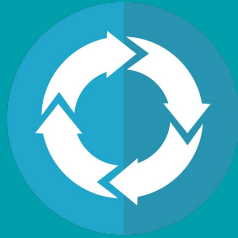


Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Iterasi pada **Dictionary**



*“ **Dictionary** merupakan salah satu tipe data yang tergolong iterable (dapat dilakukan perulangan) ”*

Dasar Perulangan Pada Dictionary

Perulangan pada **dictionary** bisa menggunakan **for loop**. Menggunakan iterasi sederhana contoh di bawah, **hanya key** pada dictionary yang akan diproses.

Contoh Code:

```

1 seller_kategori = {'sellerA':['Fashion'],
2                   'sellerB':['Fashion', 'Elektronik'],
3                   'sellerC':['Perabotan', 'Buku'],
4                   'sellerD':['Elektronik'],
5                   'sellerE':['Fashion'],
6                   'sellerF':['Perabotan', 'Elektronik']}
7
8 for i in seller_kategori:
9     print(i) #key pada dictionary

```

Notes

- i (target) bernilai key pada setiap iterasinya

Dasar Perulangan Pada Dictionary

Perulangan pada **dictionary** bisa menggunakan **for loop**. Menggunakan iterasi sederhana contoh di bawah, **hanya key** pada dictionary yang akan diproses.

Contoh Code:

```

1 seller_kategori = {'sellerA':['Fashion'],
2                     'sellerB':['Fashion', 'Elektronik'],
3                     'sellerC':['Perabotan', 'Buku'],
4                     'sellerD':['Elektronik'],
5                     'sellerE':['Fashion'],
6                     'sellerF':['Perabotan', 'Elektronik']}
7
8 for i in seller_kategori:
9     print(i) #key pada dictionary

```

```

sellerA
sellerB
sellerC
sellerD
sellerE
sellerF

```

Notes

- i (target) bernilai key pada setiap iterasinya

Dasar Perulangan Pada Dictionary - Mendapatkan Value

Pada beberapa kasus, kita membutuhkan untuk mendapatkan value pada key dibandingkan dengan key nya sendiri

Contoh Code:

```
1 for i in seller_kategori:
2     print(seller_kategori[i])
```

Notes

- **i (target)** bernilai key pada setiap iterasinya
- Untuk mendapatkan value pada dictionary kita bisa menggunakan syntax **dict_name[key]**

Dasar Perulangan Pada Dictionary - Mendapatkan Value

Pada beberapa kasus, kita membutuhkan untuk mendapatkan value pada key dibandingkan dengan key nya sendiri

Contoh Code:

```
1 for i in seller_kategori:
2     print(seller_kategori[i])
```

['Fashion']
 ['Fashion', 'Elektronik']
 ['Perabotan', 'Buku']
 ['Elektronik']
 ['Fashion']
 ['Perabotan', 'Elektronik']

Notes

- **i (target)** bernilai key pada setiap iterasinya
- Untuk mendapatkan value pada dictionary kita bisa menggunakan syntax **dict_name[key]**

Dasar Perulangan Pada Dictionary - Mendapatkan Key dan Value

Untuk lebih **mengefektifkan** kode program, kita bisa mendapatkan key dan value sekaligus pada **header line [line 2]** dengan menggunakan fungsi **.items()**

Contoh Code:

```
1 # memanggil key dan value sekaligus
2 for i in seller_kategori.items():
3     print(i) #berupa tuple

('sellerA', ['Fashion'])
('sellerB', ['Fashion', 'Elektronik'])
('sellerC', ['Perabotan', 'Buku'])
('sellerD', ['Elektronik'])
('sellerE', ['Fashion'])
('sellerF', ['Perabotan', 'Elektronik'])
```

Notes

- **i (target)** bertipe tuple berisikan 2 item, yaitu **key** dan **value** pada setiap iterasinya

Dasar Perulangan Pada Dictionary - Mendapatkan Key dan Value

Untuk lebih **mengefektifkan** kode program, kita bisa mendapatkan key dan value sekaligus pada **header line [line 2]** dengan menggunakan fungsi **.items()**

Contoh Code:

```
1 # memanggil key dan value sekaligus
2 for i in seller_kategori.items():
3     print(i) #berupa tuple
```

```
('sellerA', ['Fashion'])
('sellerB', ['Fashion', 'Elektronik'])
('sellerC', ['Perabotan', 'Buku'])
('sellerD', ['Elektronik'])
('sellerE', ['Fashion'])
('sellerF', ['Perabotan', 'Elektronik'])
```



```
1 # memanggil key dan value sekaligus
2 for key, value in seller_kategori.items():
3     print('Nilai Key', key, 'dan value adalah', value) #berupa tuple
```

```
Nilai Key sellerA dan value adalah ['Fashion']
Nilai Key sellerB dan value adalah ['Fashion', 'Elektronik']
Nilai Key sellerC dan value adalah ['Perabotan', 'Buku']
Nilai Key sellerD dan value adalah ['Elektronik']
Nilai Key sellerE dan value adalah ['Fashion']
Nilai Key sellerF dan value adalah ['Perabotan', 'Elektronik']
```

Dasar Perulangan Pada Dictionary - Mendapatkan Key dan Value

Untuk lebih **mengefektifkan** kode program, kita bisa mendapatkan key dan value sekaligus pada **header line [line 1]** dengan menggunakan fungsi **.items()**

Contoh Code:

```
1 # memanggil key dan value sekaligus
2 for i in seller_kategori.items():
3     print(i) #berupa tuple
```

```
('sellerA', ['Fashion'])
('sellerB', ['Fashion', 'Elektronik'])
('sellerC', ['Perabotan', 'Buku'])
('sellerD', ['Elektronik'])
('sellerE', ['Fashion'])
('sellerF', ['Perabotan', 'Elektronik'])
```

```
1 # memanggil key dan value sekaligus
2 for key, value in seller_kategori.items():
3     print('Nilai Key', key, 'dan value adalah', value) #berupa tuple
```

```
Nilai Key sellerA dan value adalah ['Fashion']
Nilai Key sellerB dan value adalah ['Fashion', 'Elektronik']
Nilai Key sellerC dan value adalah ['Perabotan', 'Buku']
Nilai Key sellerD dan value adalah ['Elektronik']
Nilai Key sellerE dan value adalah ['Fashion']
Nilai Key sellerF dan value adalah ['Perabotan', 'Elektronik']
```

Membuat Dictionary Baru pada For Loop

Pada beberapa kasus kita akan butuh hasil dictionary baru dari suatu iterasi. **Contoh Kasus :** Membuat dictionary baru yang berisikan seller yang menjual Category Perabotan

Contoh Code:

```
1 new_dict = {} #membuat dict kosong
2
3 for key, value in seller_kategori.items():
4     if 'Perabotan' in value: #check
5         new_dict.update({key:value})
6
7 print(new_dict)
```

```
{'sellerC': ['Perabotan', 'Buku'], 'sellerF': ['Perabotan', 'Elektronik']}
```

Dalam kasus di atas, kita bisa memanfaatkan function **.update()** untuk menambahkan key dan value yang sesuai setiap iterasinya

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Dari dictionary `seller_kategori`, buatlah variable dictionary baru (`new_seller`) yang hanya berisi seller yang menjual **produk kategori Fashion saja!**














Output:

```
7 print(new_seller)

{'sellerA': ['Fashion'], 'sellerE': ['Fashion']}
```



Loop and Iteration

- | | | | |
|--|---|---|--|
|  | Pengenalan Iterasi |  | Iterasi pada List - Indefinite Iteration |
|  | Definite Iteration |  | Pembahasan Mini Quiz Iterasi #2 |
|  | Indefinite Iteration |  | Iterasi pada Dictionary |
|  | Pembahasan Mini Quiz Iterasi #1 |  | Pembahasan Mini Quiz Iterasi #3 |
|  | Iterasi pada List - Definite Iteration |   | Iterasi pada Dataframe |
|  | One liner dan Nested Definite Iteration |  | Pembahasan Mini Quiz Iterasi #4 |

Iterasi pada **Dataframe**



“Implementasi **iterasi** sangat dibutuhkan seorang Data Scientist untuk **memanipulasi DataFrame**. Secara penerapannya iterasi pada DataFrame hampir sama dengan Dictionary.”

Dasar Perulangan pada Dataframe

Read Dataset Terlebih dulu:

```
1 import pandas as pd
2
3 df = pd.read_csv('monthly_rakamin_customer_order.csv')
4 df.tail()
```

order_month	full_name	user_id	phone	city	province	email	total_order	total_gmv	total_visit
2022-06-01	Faruq Adi	10000030	628567812319	Makassar	Sulawesi Selatan	faruq@yahoo.com	201	5490000	773
2022-06-01	Abrar Adi	10000053	628223411218	Surabaya	Jawa Timur	abrar.adi@yahoo.co.id	110	2987500	276
2022-06-01	Johan Imawan	10000054	628223412323	Balikpapan	Kalimantan Timur	johan.imawan@yahoo.com	95	2705000	258
2022-06-01	Muhammad Ammar	10000055	628223412111	Surabaya	Jawa Timur	muh.ammar@gmail.com	30	760000	103
2022-05-01	Dodo Putro Aji	10000023	628567812312	Surabaya	Jawa Timur	dodo@gmail.com	15	605000	83

Dasar Perulangan pada **Dataframe** #1

Mendapatkan nilai kolom

```
1 for i in df:  
2     print(i) #nilai i = nama kolom
```



i (target) pada perulangan **dataframe** akan bernilai nama kolom pada dataframe atau key pada dictionary

Dasar Perulangan pada **Dataframe** #1

Mendapatkan nilai kolom

```
1 for i in df:
2     print(i) #nilai i = nama kolom
```

```
order_month
user_id
full_name
phone
city
province
email
total_order
total_gmv
total_visit
```



i (target) pada perulangan **dataframe** akan bernilai nama kolom pada dataframe atau key pada dictionary

Dasar Perulangan pada Dataframe #2

Mendapatkan nilai setiap baris pada suatu kolom

```
1 #mendapatkan nilai setiap baris pada suatu kolom tertentu
2 for i in df['full_name']:
3     print(i) #nilai i = full name setiap baris
```



i (target) pada perulangan **data frame dengan nama kolom / series** akan bernilai item **full_name** pada setiap baris dataframe

Dasar Perulangan pada Dataframe #2

Mendapatkan nilai setiap baris pada suatu kolom

```
1 #mendapatkan nilai setiap baris pada suatu kolom tertentu
2 for i in df['full_name']:
3     print(i) #nilai i = full name setiap baris
```

Ahmad Budiman
Azzam Haqq
Amrina Putri
Aan Utama
Adam Adi
Budi Utomo
Bayu Aji
Bila Syahputri
Chika Ayu



i (target) pada perulangan **data frame dengan nama kolom / series** akan bernilai item **full_name** pada setiap baris dataframe

Dasar Perulangan pada Dataframe #3

Iterasi setiap baris pada Beberapa kolom

Kita bisa melakukan iterasi setiap baris dengan banyak kolom pada data frame menggunakan *function* `.iterrows()`

```
1 for index, kolom in df.iterrows():
2     user_id = kolom['user_id'] #mendapatkan value kolom user_id
3     kota = kolom['city'] #mendapatkan value kolom city
4     print(index, user_id, kota)
```

Dasar Perulangan pada Dataframe #3

Iterasi setiap baris pada Beberapa kolom

Kita bisa melakukan iterasi setiap baris dengan banyak kolom pada data frame menggunakan *function* `.iterrows()`

```
1 for index, kolom in df.iterrows():
2     user_id = kolom['user_id'] #mendapatkan value kolom user_id
3     kota = kolom['city'] #mendapatkan value kolom city
4     print(index, user_id, kota)
```

Dasar Perulangan pada Dataframe #3

Iterasi setiap baris pada Beberapa kolom

Kita bisa melakukan iterasi setiap baris dengan banyak kolom pada data frame menggunakan *function* `.iterrows()`

```
1 for index, kolom in df.iterrows():
2     user_id = kolom['user_id'] #mendapatkan value kolom user_id
3     kota = kolom['city'] #mendapatkan value kolom city
4     print(index, user_id, kota)
```

```
0 10000011 Jakarta Pusat
1 10000012 Surabaya
2 10000013 Jakarta Pusat
3 10000014 Medan
4 10000015 Jakarta Timur
5 10000016 Bandung
```


Dasar Perulangan pada Dataframe #3

Iterasi setiap baris pada Beberapa kolom

Kita bisa melakukan iterasi setiap baris dengan banyak kolom pada data frame menggunakan *function* `.iterrows()`

```
1 for index, kolom in df.iterrows():
2     user_id = kolom['user_id'] #mendapatkan value kolom user_id
3     kota = kolom['city'] #mendapatkan value kolom city
4     print(index, user_id, kota)
```

0	10000011	Jakarta Pusat
1	10000012	Surabaya
2	10000013	Jakarta Pusat
3	10000014	Medan
4	10000015	Jakarta Timur
5	10000016	Bandung

index

user_id

city

Membuat kolom baru dengan iterasi dataframe

Kita bisa memanfaatkan iterasi setiap baris pada dataframe untuk membuat kolom baru.

Studi kasus:

Buatlah kolom baru dengan nama **provider_phone** dengan ketentuan, jika 4 prefix depan sebagai berikut:

1. 0878 → XL
2. 0856 → Indosat
3. 0822 → Telkomsel
4. 0896 → Three

Membuat kolom baru dengan iterasi dataframe

Contoh Code Solusi

```

1  #membuat sebuah list kosong
2  provider_list = []
3
4  #iterasi dataframe dengan iterrows
5  for index, kolom in df.iterrows():
6      prefix = str(kolom['phone'][:5]) #get prefix
7
8      #conditional statement
9      if prefix == '62878':
10         provider = 'XL'
11     elif prefix == '62856':
12         provider = 'Indosat'
13     elif prefix == '62822':
14         provider = 'Telkomsel'
15     elif prefix == '62896':
16         provider = 'Three'
17     else:
18         provider = 'Others'
19
20     #append setiap iterasi nilai provider
21     provider_list.append(provider)
22
23 #menginisiasi kolom baru
24 df['provider_phone'] = provider_list
25

```

Membuat kolom baru dengan iterasi dataframe

Contoh Code Solusi

```

1  #membuat sebuah list kosong
2  provider_list = []
3
4  #iterasi dataframe dengan iterrows
5  for index, kolom in df.iterrows():
6      prefix = str(kolom['phone'][:5]) #get prefix
7
8      #conditional statement
9      if prefix == '62878':
10         provider = 'XL'
11     elif prefix == '62856':
12         provider = 'Indosat'
13     elif prefix == '62822':
14         provider = 'Telkomsel'
15     elif prefix == '62896':
16         provider = 'Three'
17     else:
18         provider = 'Others'
19
20     #append setiap iterasi nilai provider
21     provider_list.append(provider)
22
23 #menginisiasi kolom baru
24 df['provider_phone'] = provider_list
25

```

Proses pada For

- proses pengambilan prefix (slicing)
- conditional statement provider
- append setiap nilai provider pada setiap iterasi

Membuat kolom baru dengan iterasi dataframe

Contoh Code Solusi

```
1 #membuat sebuah list kosong
2 provider_list = []
3
4 #iterasi dataframe dengan iterrows
5 for index, kolom in df.iterrows():
6     prefix = str(kolom['phone'][:5]) #get prefix
7
8     #conditional statement
9     if prefix == '62878':
10         provider = 'XL'
11     elif prefix == '62856':
12         provider = 'Indosat'
13     elif prefix == '62822':
14         provider = 'Telkomsel'
15     elif prefix == '62896':
16         provider = 'Theree'
17     else:
18         provider = 'Others'
19
20 #append setiap iterasi nilai provider
21 provider_list.append(provider)
22
23 #menginisiasi kolom baru
24 df['provider_phone'] = provider_list
```


Membuat kolom baru dengan iterasi dataframe

Result Solusi

order_month	user_id	full_name	phone	city	province	email	total_order	total_gmv	total_visit	provider_phone
2018-02-01	10000018	Bila Syahputri	628785222218	Bandung	Jawa Barat	bila@yahoo.com	195	5,362,500	550	XL
2018-02-01	10000019	Chika Ayu	628785222219	Semarang	Jawa Tengah	chika@gmail.com	146	4,015,000	283	XL
2018-02-01	10000020	Cantika Putri Rahayu	628785222220	Bandung	Jawa Barat	cantika@gmail.co.id	21	577,500	64	XL
2018-02-01	10000021	Dimas Aji	628785222221	Semarang	Jawa Tengah	dimas@yahoo.co.id	153	4,207,500	609	XL
2018-02-01	10000022	Dika Mulya	628567812311	Palembang	Sumatra Selatan	dika@yahoo.com	88	2,420,000	294	Indosat
2018-03-01	10000023	Dodo Putro Aji	628567812312	Surabaya	Jawa Timur	dodo@gmail.com	186	5,115,000	731	Indosat

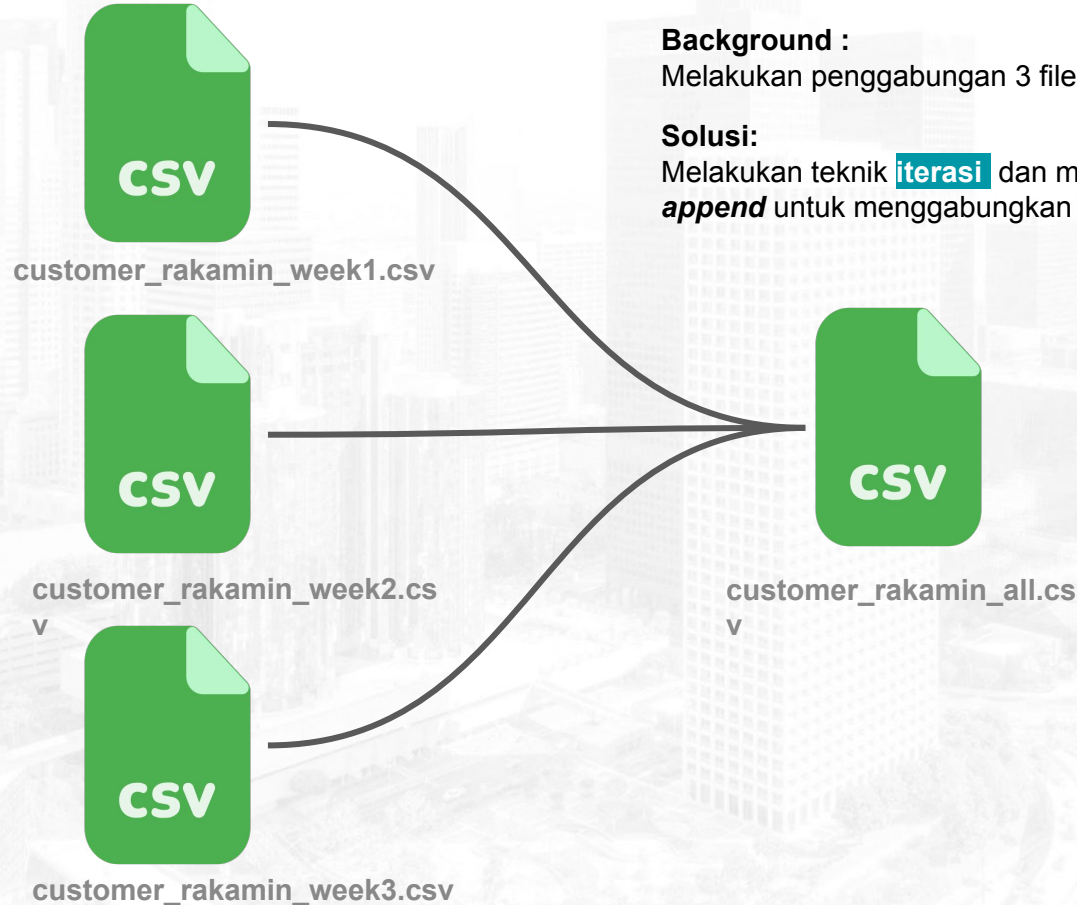
Contoh 2 penerapan iterasi

Background :

Melakukan penggabungan 3 file csv atau lebih menjadi 1 file csv

Solusi:

Melakukan teknik **iterasi** dan menggunakan fungsi **append** untuk menggabungkan 3 file csv



Contoh 2 penerapan iterasi

Background :

Melakukan penggabungan 3 file csv atau lebih menjadi 1 file csv

Solusi:

Melakukan teknik **iterasi** dan menggunakan fungsi **append** untuk menggabungkan 3 file csv

```
1 import glob
2 import pandas as pd
3
4 # get file name with start with customer_rakamin_week
5 files = glob.glob("customer_rakamin_week*")
6 print(files)

['customer_rakamin_week1.csv', 'customer_rakamin_week2.csv', 'customer_rakamin_week3.csv']
```

Package **glob** membantu kita untuk mendapatkan nama file sesuai pattern.

Penjelasan `glob.glob("customer_rakamin_week")` = mengambil semua nama file dengan pattern **customer_rakamin_week**

Contoh 2 penerapan materi iterasi

Background :

Melakukan penggabungan 3 file csv atau lebih menjadi 1 file csv

Solusi:

Melakukan teknik **iterasi** dan menggunakan fungsi **append** untuk menggabungkan 3 file csv

```
1 import glob
2 import pandas as pd
3
4 # get file name with start with customer_rakamin_week
5 files = glob.glob("customer_rakamin_week*")
6 print(files)
```

```
['customer_rakamin_week1.csv', 'customer_rakamin_week2.csv', 'customer_rakamin_week3.csv']
```

```
1 df_all = pd.DataFrame() #dataframe kosong
2
3 for i in files:
4     df_read = pd.read_csv(i) #read csv setiap file
5     df_all = df_all.append(df_read) #append
```

```
1 df_all
```

	week	province	total_order
0	1	DKI Jakarta	1200
1	1	Banten	900
2	1	Jawa Barat	1050

Contoh 2 penerapan iterasi

Background :

Melakukan penggabungan 3 file csv atau lebih menjadi 1 file csv

Solusi:

Melakukan teknik **iterasi** dan menggunakan fungsi **append** untuk menggabungkan 3 file csv

```
1 import glob
2 import pandas as pd
3
4 # get file name with start with customer_rakamin_week
5 files = glob.glob("customer_rakamin_week*")
6 print(files)
```

```
['customer_rakamin_week1.csv', 'customer_rakamin_week2.csv', 'customer_rakamin_week3.csv']
```

```
1 df_all = pd.DataFrame() #dataframe kosong
2
3 for i in files:
4     df_read = pd.read_csv(i) #read csv setiap file
5     df_all = df_all.append(df_read) #append
```

```
1 df_all
```

	week	province	total_order
0	1	DKI Jakarta	1200
1	1	Banten	900
2	1	Jawa Barat	1050

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4

Loop and Iteration



Pengenalan Iterasi



Definite Iteration



Indefinite Iteration



Pembahasan Mini Quiz Iterasi #1



Iterasi pada List - Definite Iteration



One liner dan Nested Definite Iteration



Iterasi pada List - Indefinite Iteration



Pembahasan Mini Quiz Iterasi #2



Iterasi pada Dictionary



Pembahasan Mini Quiz Iterasi #3



Iterasi pada Dataframe



Pembahasan Mini Quiz Iterasi #4



Gunakan dataset **monthly_rakamin_customer_order**! Team bisnis memiliki rule, jika dalam sebulan

- **Total order bulanan customer** > rata-rata total order bulanan semua customer di bulan tersebut (mean Order customer monthly)
- **Total GMV bulanan customer** > rata-rata **total GMV** bulanan semua customer di bulan tersebut (mean GMV customer monthly)

Maka keputusan bisnisnya **user eligible** untuk diberikan voucher discount. Ada berapa user yang menjadi **user eligible** pada **bulan Maret 2022**?

Hint: Buat kolom **mean total order monthly** dan **mean total amount monthly** (Bisa dengan cara transform), kemudian lakukan iterrows

le

Danke Schön!



Muhammad Ammar Fauzan

<https://www.linkedin.com/in/muhammad-ammar-fauzan-748883117/>