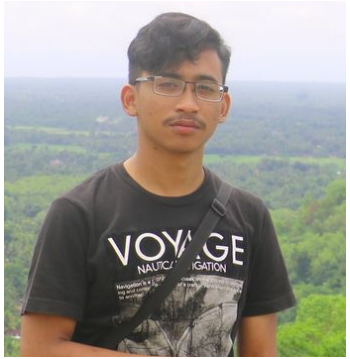
Introducing Python



Martinus Dawan

Backend Developer Kedata Indonesia

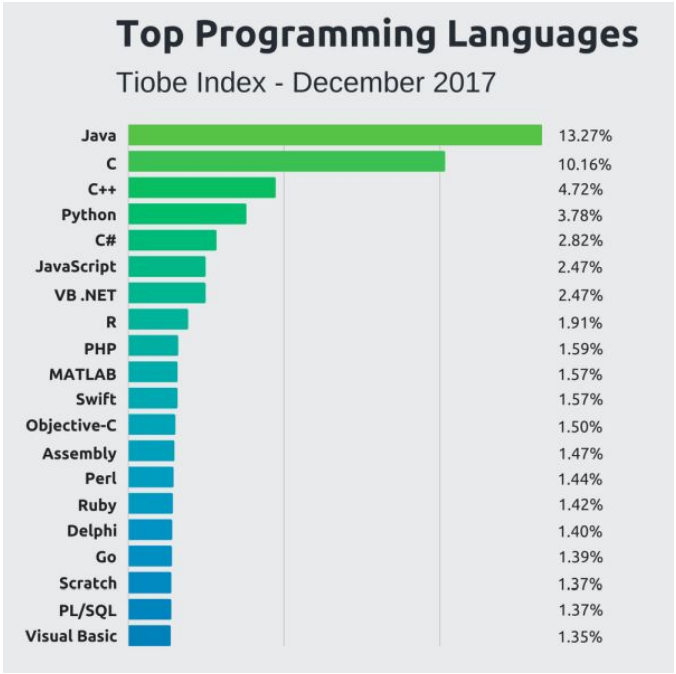
You can contact me at martinuz.dawan9@gmail.com

Overview

+

- Pengenalan Python
- Data Types
- Regular Expression

Perkembangan



(2017)

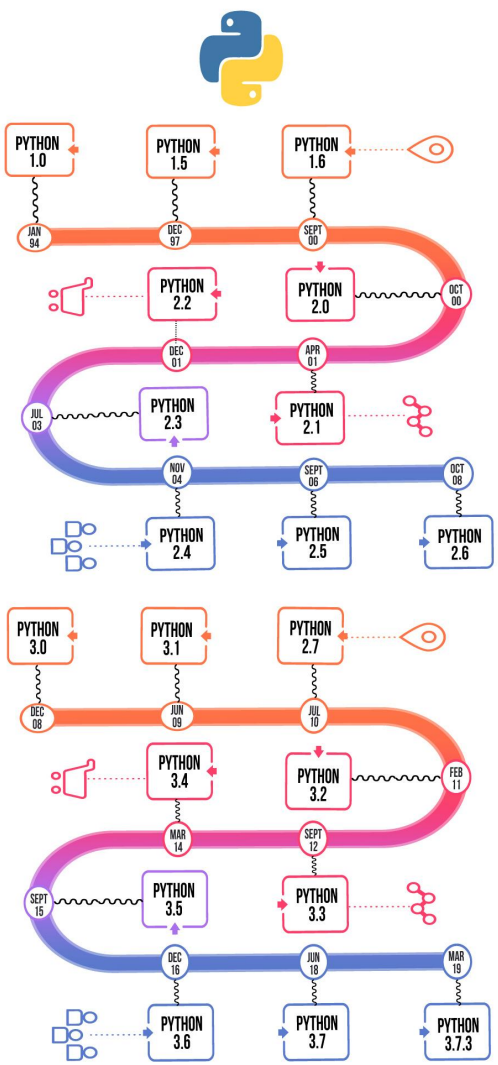
Worldwide, Oct 2019 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	29.49 %	+4.5 %
2		Java	19.57 %	-2.4 %
3		Javascript	8.4 %	+0.1 %
4		C#	7.35 %	-0.4 %
5		PHP	6.34 %	-1.2 %
6		C/C++	5.87 %	-0.4 %
7		R	3.82 %	-0.2 %
8		Objective-C	2.6 %	-0.7 %
9		Swift	2.57 %	-0.1 %
10		Matlab	1.87 %	-0.2 %

(2019)

“History of python”

Guido van Rossum

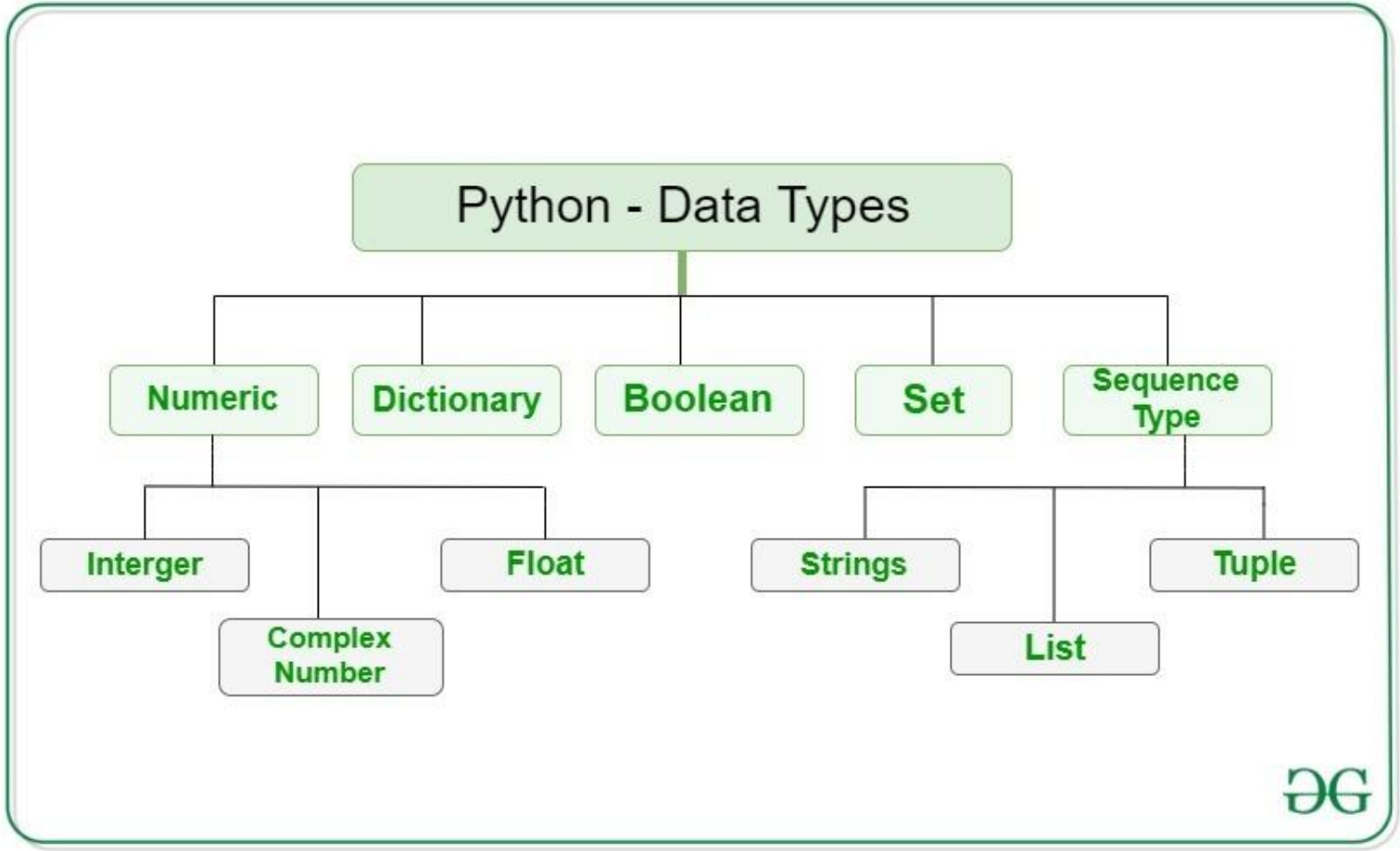

PYTHON 1.0
JAN 94
PYTHON 1.5
DEC 97
PYTHON 1.6
SEPT 00
OCT 00
PYTHON 2.0
PYTHON 2.2
DEC 01
APR 01
JUL 03
PYTHON 2.3
PYTHON 2.1
NOV 04
SEPT 06
OCT 08
PYTHON 2.4
PYTHON 2.5
PYTHON 2.6
PYTHON 3.0
DEC 08
PYTHON 3.1
JUN 09
PYTHON 2.7
JUL 10
FEB 11
PYTHON 3.2
PYTHON 3.4
MAR 14
SEPT 12
SEPT 15
PYTHON 3.5
PYTHON 3.3
DEC 16
JUN 18
MAR 19
PYTHON 3.6
PYTHON 3.7
PYTHON 3.7.3

ABC Language

```
PUT 0 IN count
FOR di, dj IN neighbours:
    IF (i+di, j+dj) in keys c:
        PUT count+c[i+di, j+dj] IN count
SELECT:
    count = 3 OR count+c[i, j] = 3:
        PUT 1 IN n[i, j]
    ELSE:
        PUT 0 IN n[i, j]
```

Mengapa Python?

- Web development (server-side)
  - Django
  - Flask
  - Tornado
- Software development
  - PyQt5
  - Tkinter
- Mathematics & data science
- Machine learning
- System scripting


Python - Data Types
Numeric
Dictionary
Boolean
Set
Sequence Type
Interger
Complex Number
Float
Strings
List
Tuple

Name	Type	Description
Integers	int	Whole numbers, such as: **3** **300** **200**
Floating point	float	Numbers with a decimal point: **2.3** **4.6** **100.0**
Strings	str	Ordered sequence of characters: **"hello"** **'Sammy'** **"2000"** **"楽しい"**
Lists	list	Ordered sequence of objects: **[10,"hello",200.3]**
Dictionaries	dict	Unordered Key:Value pairs: **{"mykey" : "value" , "name" : "Frankie"}**
Tuples	tup	Ordered immutable sequence of objects: **(10,"hello",200.3)**
Sets	set	Unordered collection of unique objects: **{"a","b"}**
Booleans	bool	Logical value indicating **True** or **False**

Numeric

```
>>> print(123 + 1)
124

>>> a = 123 + 1
124

>>> type(a)
<class 'int'>


>>> print(12.1 + 1)
13.1

>>> type(12.1 + 1)
<class 'float'>
```

Sequence

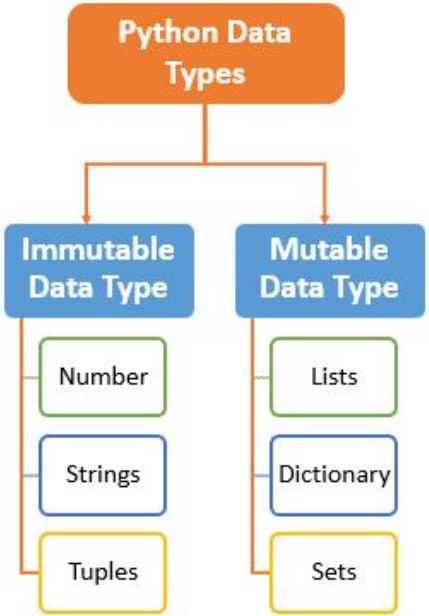
```
>>> tipe_string = "I am a string."

>>> print(tipe_string)
I am a string.

>>> type(tipe_string)
<class 'str'>


# Tuple

>>> angka = (1, 2, 3)
>>> angka
(1, 2, 3)
>>> type(angka)
<class 'tuple'>
```


Python Data Types
Immutable Data Type
Mutable Data Type
Number
Strings
Tuples
Lists
Dictionary
Sets

Lists

Coding!

Source code: https://github.com/martinusdawan/gino-satisfying

Dictionary

Coding!

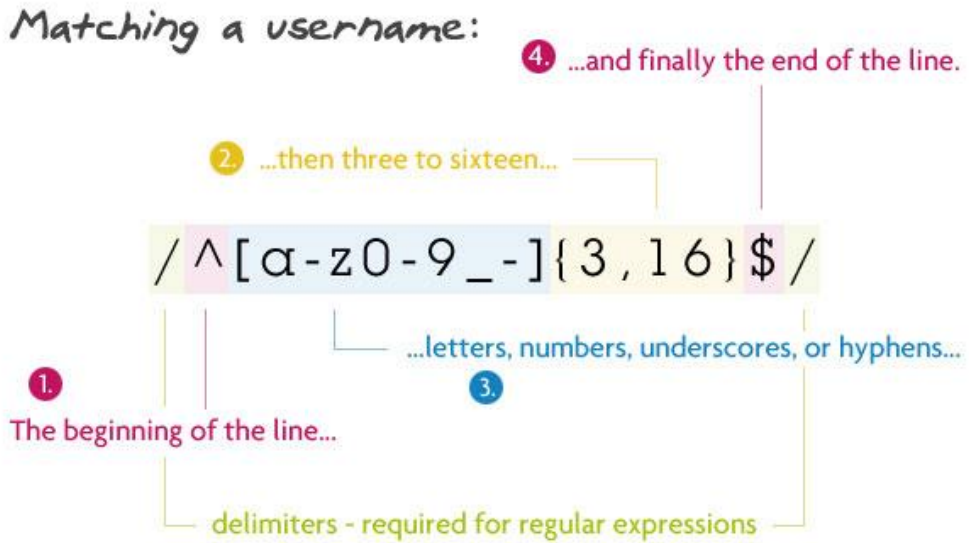
Source code: https://github.com/martinusdawan/gino-satisfying

Regex (Regular Expression)

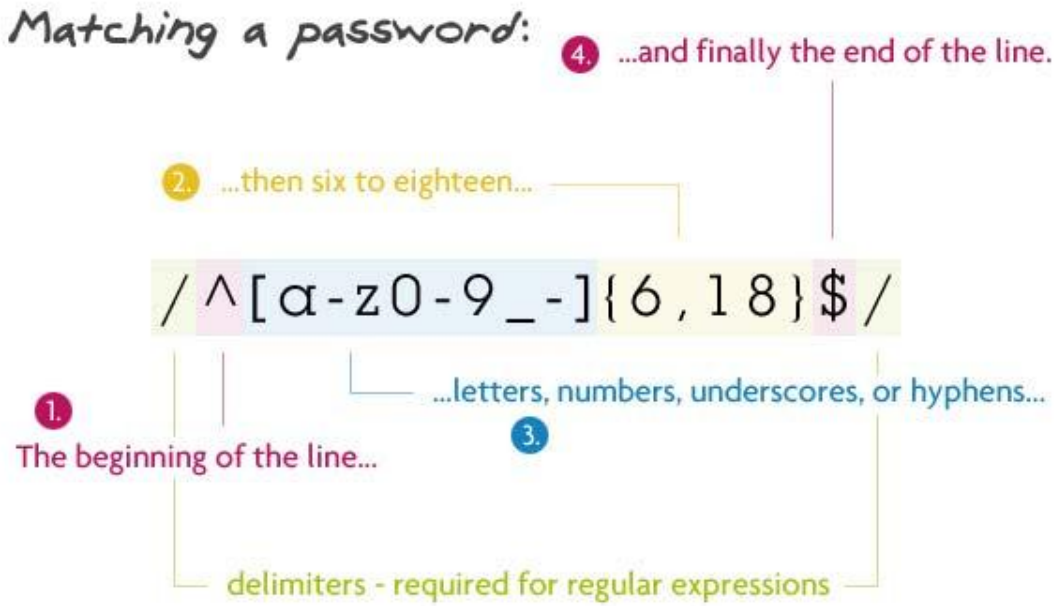
- Pencocokan Username, Password dan Email
- Pencocokan URL

“Dalam komputasi, regular expressions menyediakan sarana yang ringkas dan fleksibel untuk mengidentifikasi string teks menarik, seperti karakter tertentu, kata-kata atau pola karakter.”

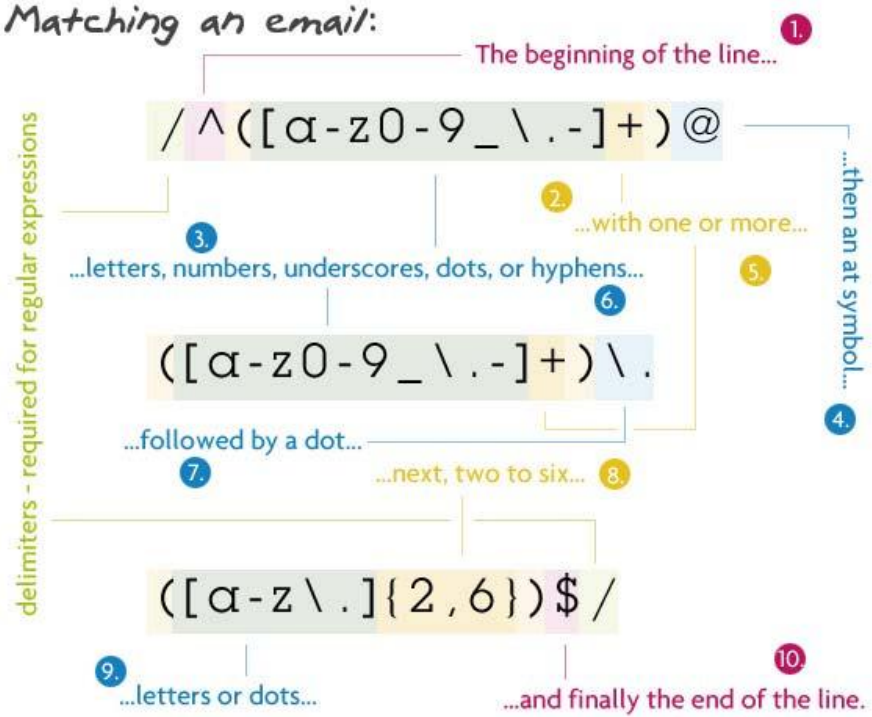
Pencocokan Username



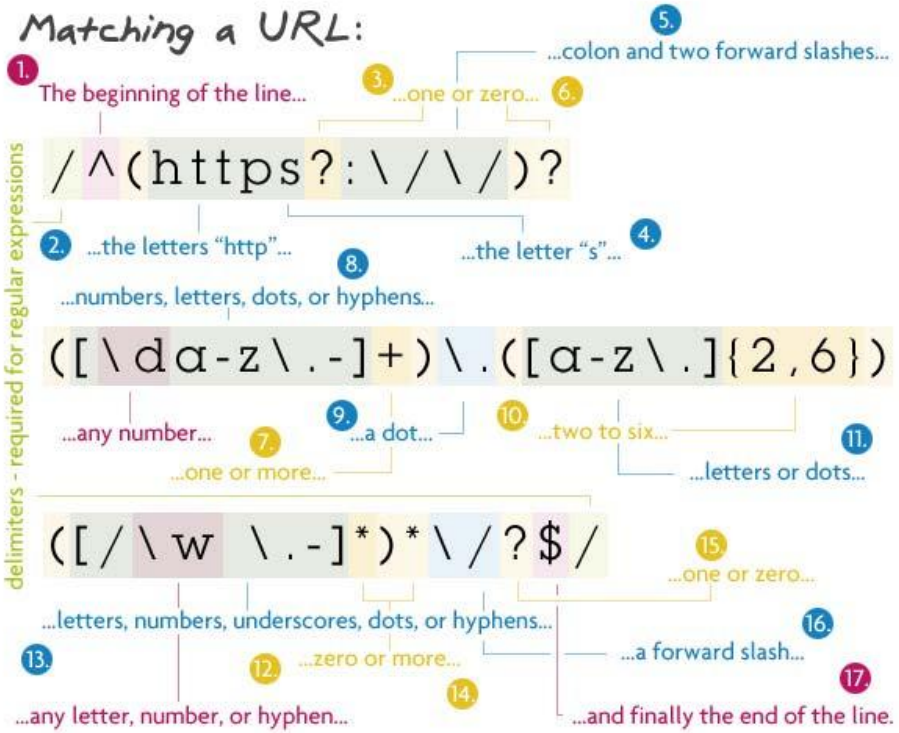
Pencocokan Password



Pencocokan Email



Pencocokan URL



```python
import re

# 1
username = 'th1s1sw*ayt00l0ngt'
types_username = re.findall(r'^[a-zA-Z0-9]+(?:[_ -]?[a-zA-Z0-9])\w{3,14}$', username)
print(types_username)

# 2
username = 'th1s123'
types_username = re.findall(r'^[a-zA-Z0-9]+(?:[_ -]?[a-zA-Z0-9])\w{3,14}$', username)
print(types_username)

# 3
password = 'aBc45DSD_sdf'
types_password = re.findall(r'^[a-zA-Z0-9]\w{3,14}$', raw_string)
print(types_password)

# 4
password = '\tth1s1234'
types_password = re.findall(r'^[a-zA-Z0-9]\w{3,14}$', raw_string)
print(types_password)


# Output 1
[]

# Output 2
['th1s123']

# Output 3
['aBc45DSD_sdf']

# Output 4
[]
```

```
import re

email1 = 'john@123doe.abc'
types_email = re.findall(r'^\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,3}$', email1)

print(types_email)

email1 = 'john@doe.com'
types_email = re.findall(r'^\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,3}$', email1)

print(types_email)

# Output
>>> ['']
>>> ['john@doe.com']

# URL
url_string = 'https://www.youtube.com/watch?v=9U6meqmEsrY'

text = re.sub(r'((www\.[^\s]+)|(https?://[^\s]+))', 'terhapus', url_string)

print(text)

# output
>>> terhapus
```

Thanks!

Any questions?