



Basic Python

<https://github.com/kaopanboonyuen/CS101>

Reference:

1. <https://stanfordpython.com>
2. <https://ocw.mit.edu/courses/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/>
3. <https://stanford.edu/~schmit/cme193/>
4. <https://github.com/suneelpatel/Python-for-Beginners/tree/master>
5. <https://data-flair.training/blogs/python-career-opportunities/>
6. <https://geekflare.com/google-colab/>
7. <https://cs231n.github.io/python-numpy-tutorial/>

About Me



Kao
Panboonyuen

kao-panboonyuen 

AI Research Scientist

Name: Teerapong Panboonyuen

Contact: teerapong.pa@chula.ac.th
panboonyuen.kao@gmail.com

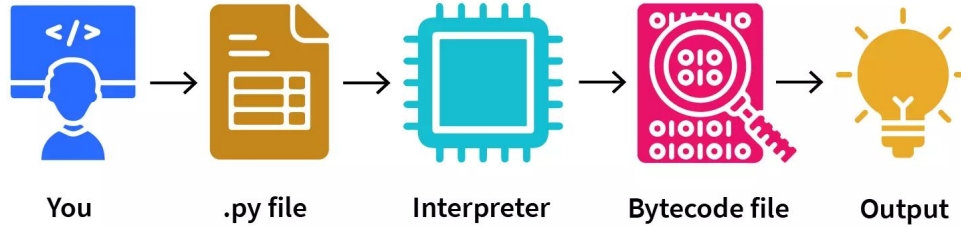
Education: Ph.D. in Computer Eng., Chula

Position: AI Team Lead, MARS

PostDoc, Chula

Interests: Computer Vision, Deep Learning
Machine Learning

What is PYTHON ?

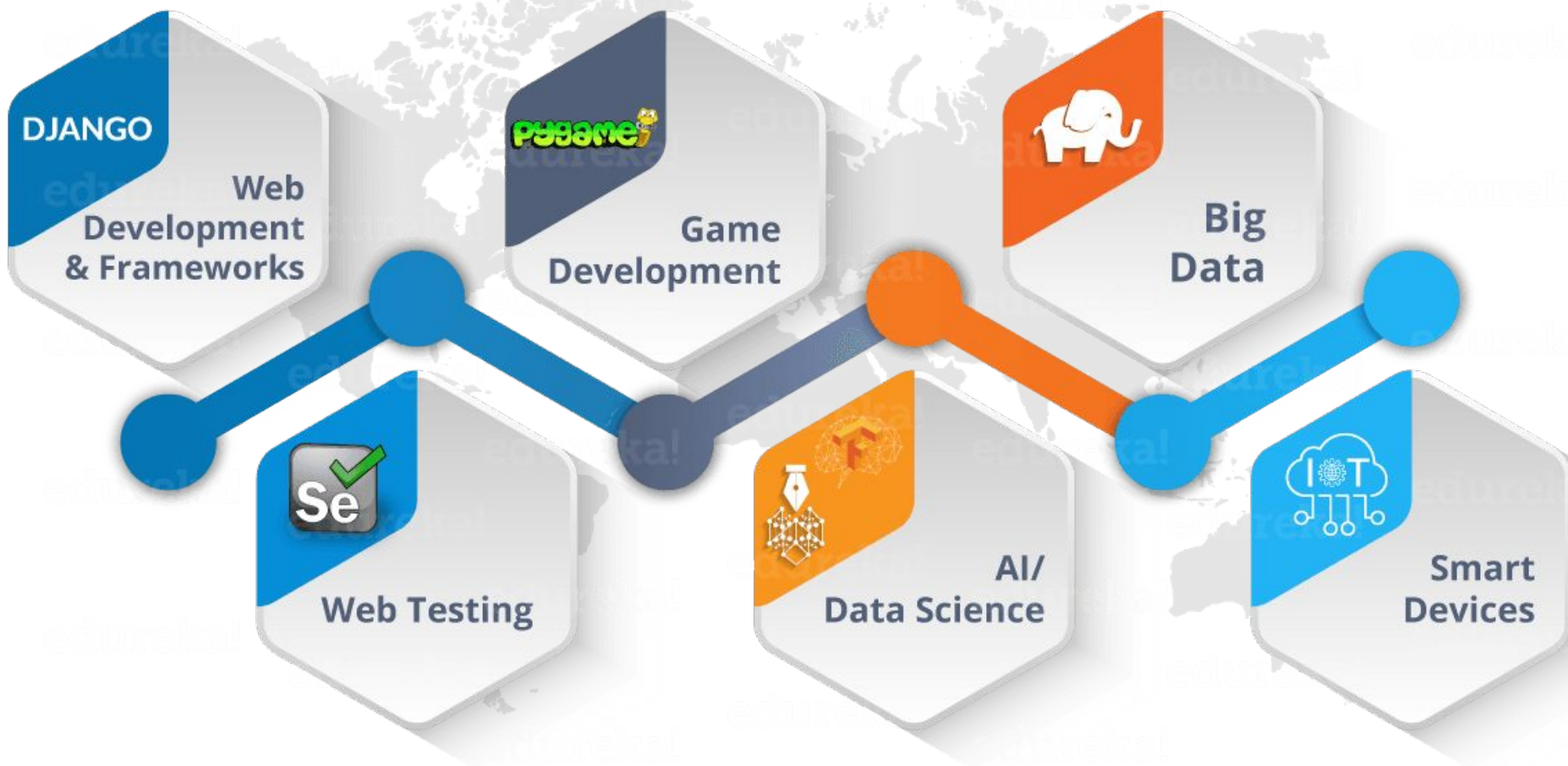


The process to generate the output :

- Write a high-level python code.
- Save the code in .py file.
- Interpret the code.
- It will generate a bytecode file.
- The output will get printed on the screen.

Python Career Opportunities



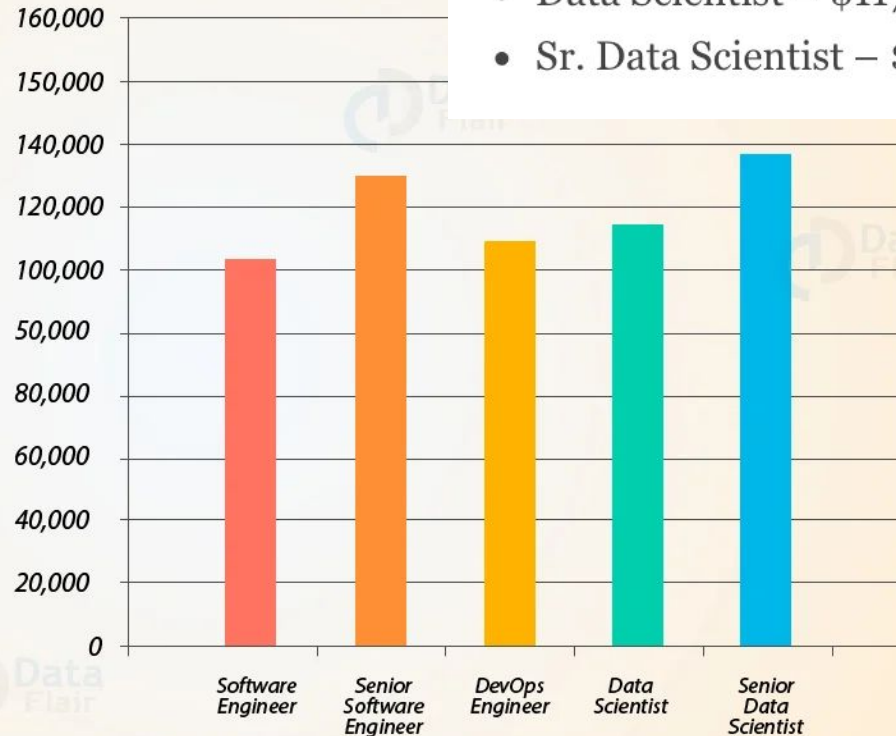


Python Salary

- Software Engineer – \$103,035/yr
- Sr. Software Engineer – \$129,328/yr
- DevOps Engineer – \$115,666/yr
- Data Scientist – \$117,345/yr
- Sr. Data Scientist – \$136,633/yr



**Python
Payscale**





WHY



Python

01

Simplicity

02

Large Community

03

High
Demand-Supply Ratio

04

Large
Number of Frameworks

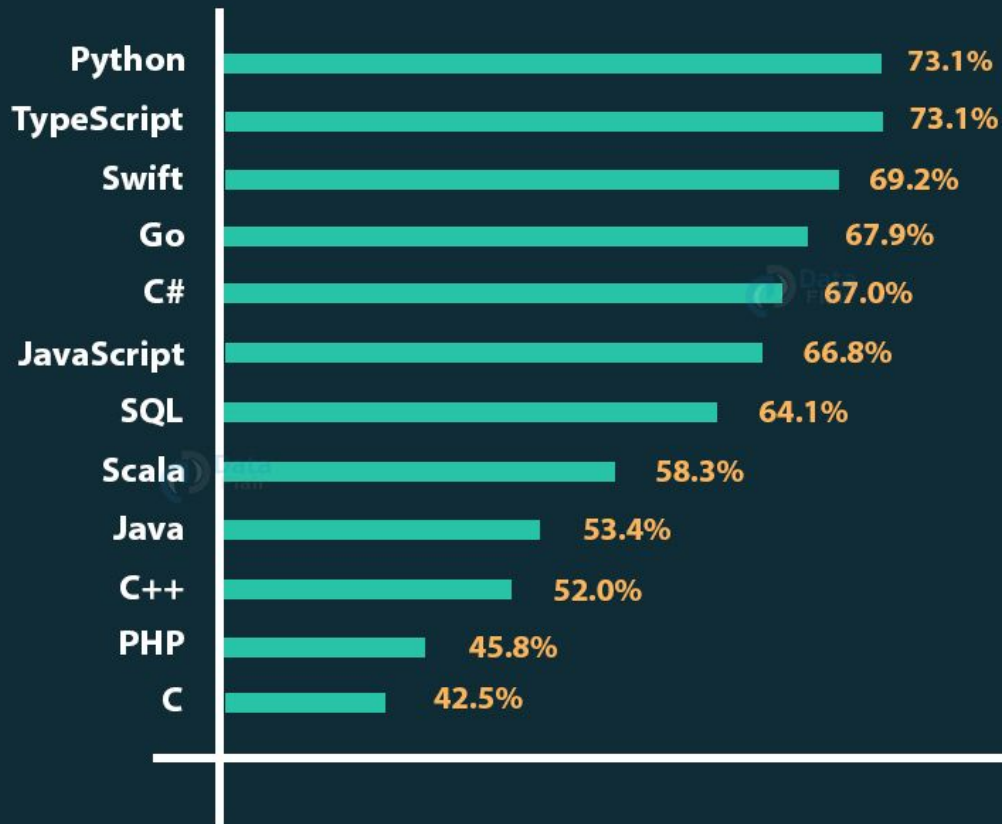
05

Chosen Language
for AI and ML

06

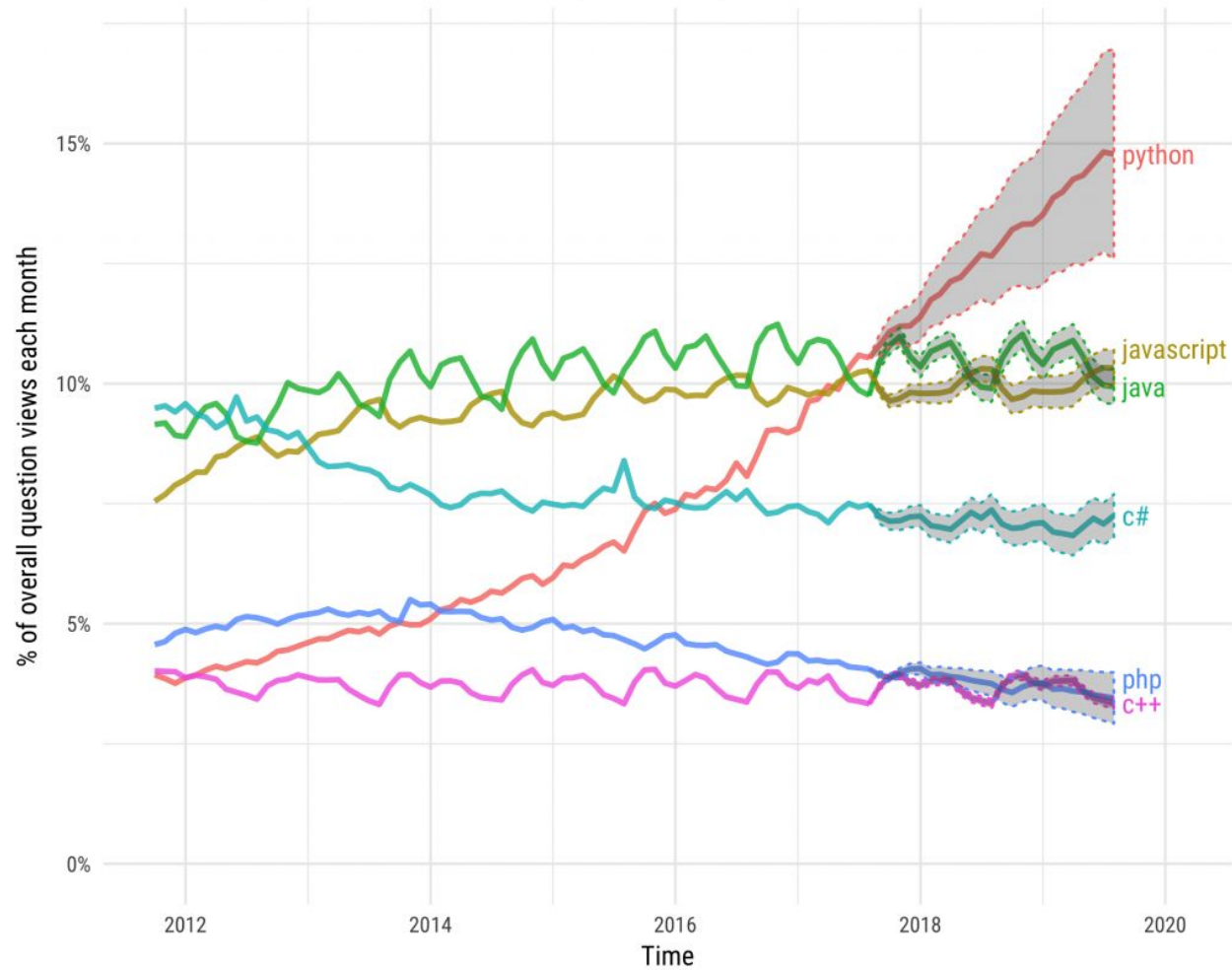
Make your own DIYs

Most Popular Languages



Projections of future traffic for major programming languages

Future traffic is predicted with an STL model, along with an 80% prediction interval.



5 REASONS TO LEARN PYTHON

GENTLE LEARNING CURVE

01

Cited as one of the easiest programming languages for beginners to learn.

CROSS- FUNCTIONAL

02

As a "general purpose language", Python is used for web development and data science.

WORKS ON IoT DEVICES

03

Python can be used to build a remote controlled car, a mini robot, or a beer keg monitor.

ESTABLISHED COMMUNITY

04

Python was released in 1991, and is open source, so it's always improving and changing.

HIGH SALARY

05

A Python programmer's salary is "better than the average software developer."

Top Companies using Python



NETFLIX



YAHOO!

moz://a

amazon



UBER

shutterstock



SurveyMonkey

JPMorganChase

SendGrid



IBM

trivago



9GAG

yelp

Udemy



reddit



Prezi

Vine

asana

freelancer.com
Outsourcing for Small Business

Bitbucket



NOKIA

@mailgun
Email Automation



Bank of America



Spotify



redis

PANDORA

hi hike

DISQUS

MIT

Massachusetts
Institute of
Technology

Learn Basics Concept of Python



A portrait of Mark Zuckerberg, looking directly at the camera with a neutral expression. He has short, wavy brown hair and is wearing a dark blue t-shirt. The background is a blurred office environment with warm, golden light from overhead fixtures.

MARK

CREATED facebook

What is Python?

- Created in 1991 by Guido Van Rossum
- Named after Monty Python's Flying Circus
- Is an interpreted, Object-Oriented, High Level Programming Language with dynamic semantics.
- Gained popularity because of its clear syntax and readability

Why Learn Python?

Python's syntax is very easy to understand. The lines of code required for a task is less compared to other languages

```
print ("Welcome To Python  
Programming!")
```

Let's see, What experts have to say about Python:

- Simple and Easy to learn
- Free and open source
- High Level Language
- Support different programming paradigm
 - Object Oriented
 - Procedure Oriented
 - Functional Oriented
 - Imperative Oriented
- Portable

Google Colaboratory



What is Google Colab?

- Google Colaboratory, or "**Colab**" as most people call it, is a **cloud-based Jupyter notebook** environment.
- It runs in your web browser (you can even run it on your favorite Chromebook) and lets anyone with internet access experiment with machine learning and coding for artificial intelligence.

Best Features of Google Colab

Free Colab users get chargeless access to GPU and TPU runtimes for up to **6 hours**. Its GPU runtime comes with **Intel Xeon CPU @2.20 GHz, 13 GB RAM, Tesla K80 accelerator, and 12 GB GDDR5 VRAM**.

The TPU runtime consists of an **Intel Xeon CPU @2.30 GHz, 13 GB RAM**, and a cloud TPU with 180 teraflops of computational power.

With Colab **Pro or Pro+**, you can commission **more CPUs, TPUs, and GPUs for more than 12 hours**.

Notebook Sharing

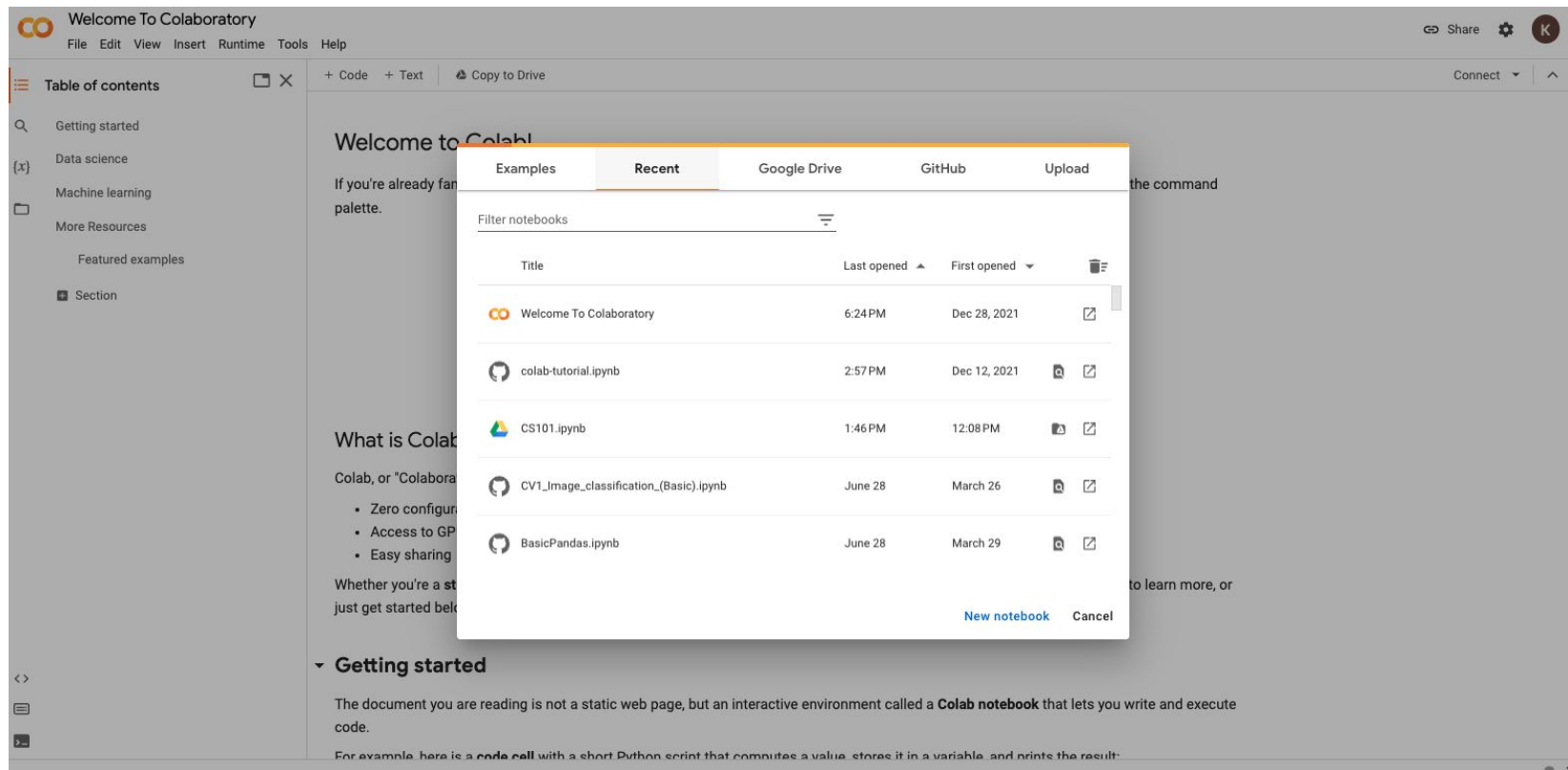
Python code notebook has never been accessible before Colab. Now, you can create shareable links for Colab files that are saved on your Google Drive.

Now, share the link with the collaborator who wants to work with you. Moreover, you can also invite programmers to work with you using Google emails.

Differences Between Google Colab and Jupyter Notebook

Code Doc Features	Google Colaboratory	Jupyter Notebook
Instant code file viewing	Yes	No
Code doc sharing	Yes	No
Installed libraries	Yes	No
Cloud hosting	Yes	No
Syncing files	Yes	No

Access GitHub, Local Files, Google Drive, AWS, etc



The screenshot displays the Google Colaboratory web interface. A modal window is open, showing a list of recent notebooks. The modal has tabs for 'Examples', 'Recent' (selected), 'Google Drive', 'GitHub', and 'Upload'. Below the tabs is a search bar labeled 'Filter notebooks'. The notebook list includes columns for 'Title', 'Last opened', 'First opened', and icons for opening or deleting. The background shows the 'Welcome To Colaboratory' page with a sidebar menu and a main content area.

Title	Last opened	First opened	Icons
Welcome To Colaboratory	6:24 PM	Dec 28, 2021	[Open] [Delete]
colab-tutorial.ipynb	2:57 PM	Dec 12, 2021	[Open] [Delete]
CS101.ipynb	1:46 PM	12:08 PM	[Open] [Delete]
CV1_Image_classification_(Basic).ipynb	June 28	March 26	[Open] [Delete]
BasicPandas.ipynb	June 28	March 29	[Open] [Delete]

Buttons at the bottom of the modal: [New notebook](#) and [Cancel](#).

<https://colab.research.google.com/>

Google Colaboratory



Python Variables

- A Python variable is a reserved memory location to store values. In other words, a variable in a python program gives data to the computer for processing.
- Variables are containers for storing data values. Unlike other programming languages, Python has **no command for declaring a variable**.

```
seconds_in_a_day = 24 * 60 * 60  
seconds_in_a_day
```

86400

Python Variable Name Rules

- Must begin with a **letter** (**a - z, A - B**) or **underscore** (**_**)
- Other characters can be letters, numbers or **_**
- **Case Sensitive**
- Can be any (reasonable) length.
- There are some reserved words which you cannot use as a variable name because Python uses them for other things.

Keywords

ou can't use a keyword as variable name, function name or any other identifier name. Here is the list of keywords in python.

Keywords in Python				
False	class	<u>finally</u>	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	<u>elif</u>	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Python Tokens

- Keyword
 - Python keywords are special reserve keywords
- Identifiers
 - Names that the programmer defines
- Literals
 - Values classified by types: e.g., numbers, truth values, text
- Operators
 - Symbols that operate on data and produce results

Python Tokens: Identifiers

- Identifiers are the name to **identify** a variable, function, class or an object
- RULES defined for naming an identifiers:
 - No special character **except underscore** (`_`) can be used as an identifier
 - Keywords should not be used as an identifiers
 - Python is case sensitive, i.e. **Var** and **var** are two different identifier
 - First character of an identifier can be character, underscore(`_`) but not digit.

Python Tokens: Literals

Literals are data given in a variable or constant

- String Literals
 - Formed by enclosing a text in the quotes
 - Both single and double quotes can be used
- Numeric Literals
 - Int | long | Float | Complex
- Boolean Literals
 - Can have only two values: True | False
- Special Literals
 - Python has one special literal: None
 - Used to specify to the field that is not created

```
# Integer
my_integer = 42
print("Integer:", my_integer)

# Float
my_float = 3.14
print("Float:", my_float)

# Complex
my_complex = 2 + 3j
print("Complex:", my_complex)
```

Data Structure (Data Types in Python)

- Immutable Data
 - Numeric : There are three numeric types in Python:
 - **Int** : Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.
 - **Float** : Float, or "floating point number" is a number, positive or negative, containing one or more decimals.
 - **Complex** : Complex numbers are written with a "j" as the imaginary part:
 - String : Combination of characters within the single or double quotes or it's a continuous series of characters surrounded by single or double quotes
 - Tuples : A tuple is a collection which is ordered and immutable (unchangeable). In Python tuples are written with round brackets.

Data Structure (Data Types in Python) (cont.)

- **Mutable Data**
 - **List** : A list is a collection which is ordered and changeable. In Python lists are written with square brackets. []
 - **Set** : A set is a collection which is unordered and unindexed. In Python sets are written with curly brackets { }
 - **Dictionary** : A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets { }, and they have keys and values in each items.

Conditional Statement

- Python Conditional Statement : If ... Else Statement
- Python Conditional Operators (conditions) and If statements
- Python supports the usual logical conditions from mathematics:
 - **Equals:** $a == b$
 - **Not Equals:** $a != b$
 - **Less than:** $a < b$
 - **Less than or equal to:** $a <= b$
 - **Greater than:** $a > b$
 - **Greater than or equal to:** $a >= b$

Flow Control

There are six basic flow controls used in Python programming:

- if
- for
- while
- break
- continue
- pass

```
# Example of a for loop with if statement and break
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

for number in numbers:
    if number > 5:
        print("Breaking the loop because number is greater than 5.")
        break
    print("Number:", number)
```

Loops in Python

- **while loop:** Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.
- **for loop :** Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.
- **nested loops:** You can use one or more loop inside any another while, for or do..while loop.

```
# Example of a while loop
count = 0
while count < 5:
    print("Count:", count)
    count += 1
```

```
# Example of a for loop
numbers = [1, 2, 3, 4, 5]
for number in numbers:
    print("Number:", number)
```

```
# Example of nested loops
rows = 3
columns = 3

for i in range(rows):
    for j in range(columns):
        print(f"({i}, {j})")
```

Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

```
def calculate_square(num):  
    """Function to calculate the square of a number."""  
    square = num ** 2  
    return square  
  
# Calling the function  
result = calculate_square(5)  
print("Square:", result)
```

Exception Handling:

The **"try"** block lets you test a block of code for errors.

The **"except"** block lets you handle the error.

The **"finally"** block lets you execute code, regardless of the result of the try- and except blocks.

```
def divide_numbers(a, b):  
    """Function to divide two numbers."""  
    try:  
        result = a / b  
        print("Division result:", result)  
    except ZeroDivisionError:  
        print("Error: Division by zero is not allowed.")  
    finally:  
        print("This will always execute, regardless of exceptions.")  
  
# Calling the function  
divide_numbers(10, 2)  
divide_numbers(10, 0)
```



```
Division result: 5.0  
This will always execute, regardless of exceptions.  
Error: Division by zero is not allowed.  
This will always execute, regardless of exceptions.
```

Submission Instructions

- Submitting through teerapong.pa@chula.ac.th
- Download your ipynb (from Colab) as a .ipynb file. With your Notebook pulled up in Instabase, go to Open With > Download. This will automatically download your Notebook as a .ipynb file.
- Then, **compress** the .ipynb file as **.zip** by naming it
- "Student ID_Name_LastName_PythonAssignment01.zip" and send it to teerapong.pa@chula.ac.th



<https://www.eng.chula.ac.th/th/20535>



ดาวน์โหลดได้ที่

<https://www.cp.eng.chula.ac.th/books/python101/>

<https://www.youtube.com/watch?v=U2l1xgpVsu&list=PL0ROnACzUGB4ieaQndKybT9xyoq2n9NGq>

2110101

**Computer Programming
(Python)**

คณะวิศวกรรมศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย

CHULA **ENGINEERING**
Foundation toward Innovation

 **python**



ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
จุฬาลงกรณ์มหาวิทยาลัย

//letsDo
Coding