



Python basic

ITFORVN - Python 1

Python environment

02

1. Python on Linux, Windows and macOS
2. Python shell
3. Python modules
4. Virtual environment
5. Basic scripting

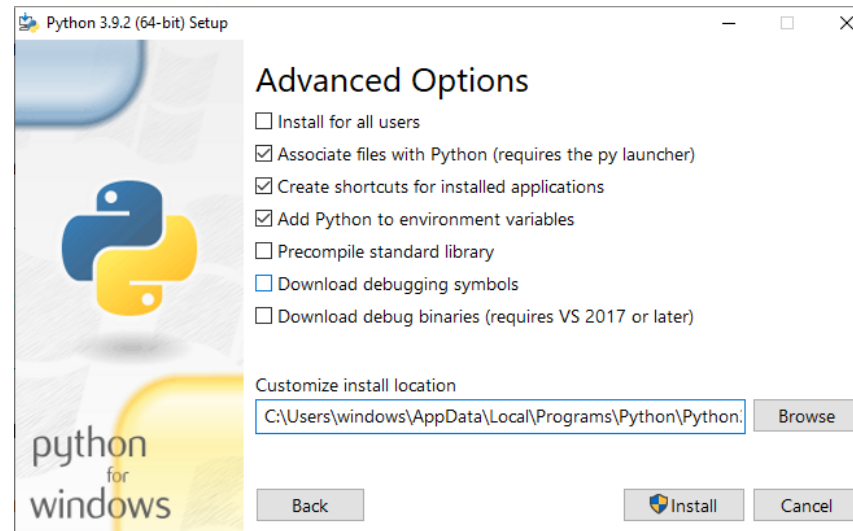
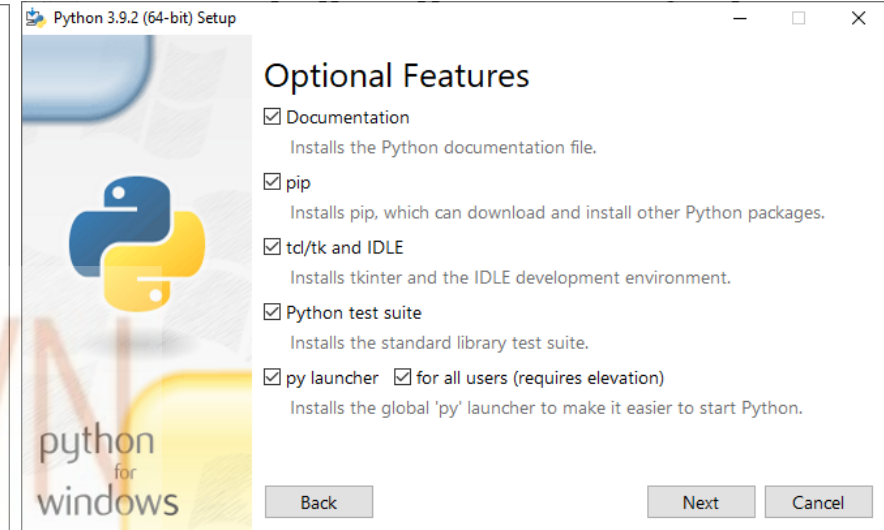
1. Python on Linux, Windows and macOS

- Linux:
 - Pre-installed trên hầu hết các Linux distro – Cpython.
 - Nhiều ứng dụng, system utilities, GUI cần Python để chạy.
 - XÓA hoặc REMOVE Python có thể gây “kết quả” nghiêm trọng.
- Windows:
 - Python for Windows.
 - <https://www.python.org/downloads/windows/>
- macOS:
 - Pre-installed



1. Install Python on Windows 10

- Download Python 3
- Run executable:
 - Add Python to PATH
 - Install pip
 - Associate files with Python
 - Add Python to environment variables
- Test:
 - `py`
 - `pip`



2. Python shell

- Execute SINGLE PYTHON COMMAND and display the result.
- Còn được biết đến với tên REPL: Read, Evaluate, Print, Loop.
- Nhận diện bằng prompt >>>
- Thoát Python shell: **Ctrl+D** hoặc `quit()`

```
PS C:\Program Files\Terminus> py
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> copyright
Copyright (c) 2001-2021 Python Software Foundation.
All Rights Reserved.

Copyright (c) 2000 BeOpen.com.
All Rights Reserved.

Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.

Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.
>>> █
```

```
~/Projects > python3
Python 3.8.5 (default, Jul 28 2020, 12:59:40)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> copyright
Copyright (c) 2001-2020 Python Software Foundation.
All Rights Reserved.

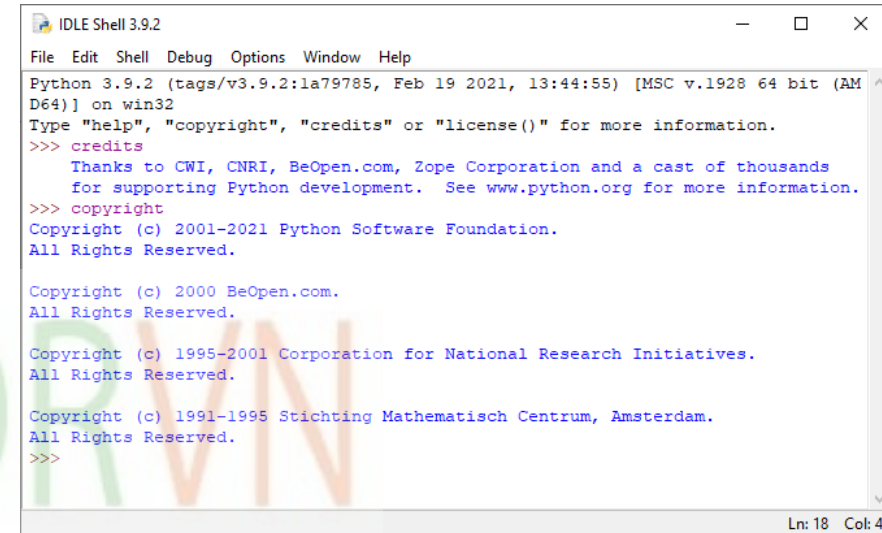
Copyright (c) 2000 BeOpen.com.
All Rights Reserved.

Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.

Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.
>>>
```

2. Python shell - IDLE

- IDLE – một chương trình được cài đặt cùng với Python trên Windows, macOS – là một chương trình “Python shell” hỗ trợ việc coding Python.
- IDLE (không phải IDE – Interactive Development Environment) – tên được lấy cảm hứng từ Eric Idle, người sáng lập nhóm Monty Python.
- Trên Ubuntu, cài đặt IDLE như sau:
 - Python 2.7: `sudo apt install idle-python2.7`
 - Python 3.8: `sudo apt install idle-python3.8`

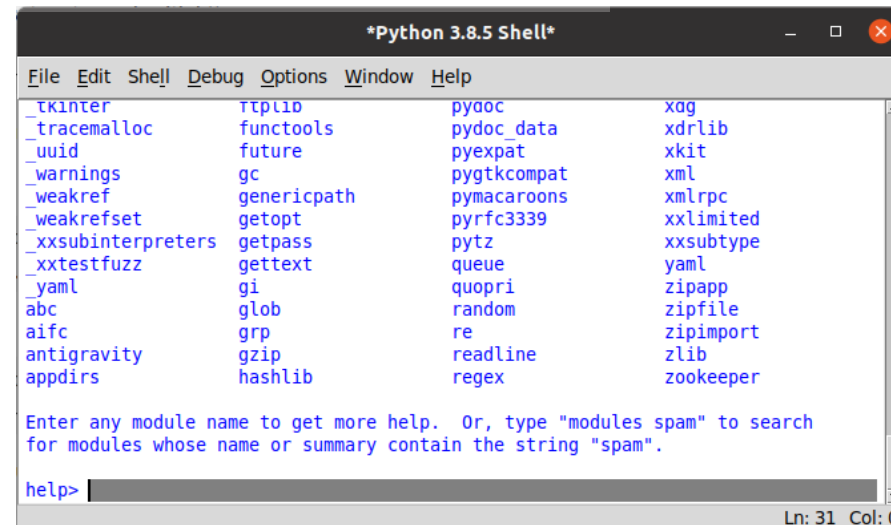


```
IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> credits
Thanks to CWI, CNRI, BeOpen.com, Zope Corporation and a cast of thousands
for supporting Python development. See www.python.org for more information.
>>> copyright
Copyright (c) 2001-2021 Python Software Foundation.
All Rights Reserved.

Copyright (c) 2000 BeOpen.com.
All Rights Reserved.

Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.

Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.
>>>
```



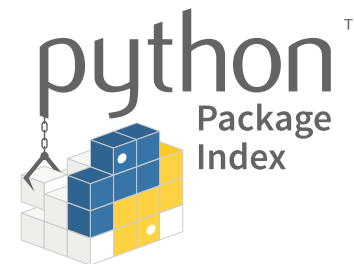
```
*Python 3.8.5 Shell*
File Edit Shell Debug Options Window Help
_tkinter      rtpdb         pydoc         xag
_tracemalloc  functools     pydoc_data    xdrllib
_uuid         future        pyexpat       xkit
_warnings     gc            pygtkcompat   xml
_weakref      genericpath   pymacaroons   xmlrpc
_weakrefset   getopt        pyrfc3339     xxlimited
_xxsubinterpreters getpass       pytz          xxsubtype
_xxtestfuzz   gettext       queue         yaml
_yaml         gi            quopri        zipapp
abc           glob          random        zipfile
aifc          grp           re            zipimport
antigravity   gzip          readline      zlib
appdirs       hashlib       regex         zookeeper

Enter any module name to get more help. Or, type "modules spam" to search
for modules whose name or summary contain the string "spam".

help>
```

3. Python modules

- Library: hay gọi là thư viện, là các “code” mang tính tái sử dụng, được xây dựng sẵn, và thường được xây dựng cho một mục đích.
- Python libraries thường được gọi dưới tên “modules”.
- Standard library: được cài đặt cùng với Python interpreter và sẵn sàng để được sử dụng, thường được document rất tốt (cụ thể, dễ hiểu, chi tiết).
- Python modules được viết bằng mã Python, hoặc binary.
- Package: các code được xem là hay/chuẩn, được đóng gói theo các rule nhất định, bao gồm các metadata, installation parameter, directory layout phù hợp. Các package có thể được chia sẻ để cài đặt sử dụng.
- The Python Package Index (PyPI) is a repository of software for the Python programming language: <https://pypi.org/>



3. Python modules – install modules

- Python modules thường được chia sẻ và phân phối trên PyPI, và được cài đặt dễ dàng thông qua công cụ **pip** (*included in Python 2.7 and 3.4+*)

pip install package_name

- Với các packages không có trên PyPI, và được phân phối dưới dạng package archive, dùng cách cài đặt manual:

```
tar xf packages-1.0.tar.gz #extract package
cd packages-1.0
python setup.py install    #install package
```

- easy_install**: một cách khác để cài đặt Python modules. **KHÔNG SỬ DỤNG**

	pip	easy_install
Installs from Wheels	Yes	No
Uninstall Packages	Yes (<code>python -m pip uninstall</code>)	No
Dependency Overrides	Yes (Requirements Files)	No
List Installed Packages	Yes (<code>python -m pip list</code> and <code>python -m pip freeze</code>)	No
PEP 438 Support	Yes	No
Installation format	'Flat' packages with <code>egg-info</code> meta-data.	Encapsulated Egg format
sys.path modification	No	Yes
Installs from Eggs	No	Yes
pylauncher support	No	Yes [1]
Multi-version installs	No	Yes
Exclude scripts during install	No	Yes
per project index	Only in virtualenv	Yes, via setup.cfg

3. Python modules – where is it stored?

- Tùy vào từng platform, modules sẽ được lưu tại các đường dẫn khác nhau.
- Các cách xác định đường dẫn lưu trữ modules:
 - Built-in
 - `sys.path`: path mà Python sẽ tìm các modules
 - `__file__`: path chính xác của module đang được gọi. Ví dụ: `xml.__file__`
 - `help()`
- Linux:
 - From Debian package manager: `/usr/lib/pythonx.y/dist-packages`
 - From `pip` & `easy_install`: `/usr/local/lib/pythonx.y/dist-packages`

3. Python modules – usage

- Importing module:

```
import module_name  
  
module_name.some_function()
```

- Using from ... Import ...

```
from module_name import some_function  
  
some_function()
```

- Aliasing modules

```
import module_name as my_module  
  
my_module.some_function()
```

- Module import priority:

1. Built-in modules
2. System path

```
#!/usr/bin/python3  
  
import sys  
import os  
  
### Built-in modules  
print(sys.modules)  
  
### Modules path  
print(sys.path)
```

4. Virtual environment

- Virtual environment - “isolated environment”, tách biệt các modules, dependencies nhằm tạo ra một môi trường riêng biệt dành riêng cho mỗi “project” sử dụng.
- Virtual environment lưu giữ và tổ chức toàn bộ “Python” trong 1 thư mục tùy chọn, không ảnh hưởng đến “Python system” mà hệ điều hành đang sử dụng.
- Các bước sử dụng “virtual environment”:
 - apt install python3-venv
 - python3 -m venv *environment_name*
 - source *environment_name/bin/activate*: active environment
 - deactivate: hủy active environment

“activate” thực hiện việc PREPEND bin directory trong “environment_name” vào current \$PATH và chỉnh sửa \$PS1 cho prompt

```
root@ubuntu:/home/ubuntu/Projects/Python-1/venv# tree -L 2 .
.
├── bin
│   ├── activate
│   ├── activate.csh
│   ├── activate.fish
│   ├── Activate.ps1
│   ├── easy_install
│   ├── easy_install-3.8
│   ├── pip
│   ├── pip3
│   ├── pip3.8
│   ├── python -> python3
│   └── python3 -> /usr/bin/python3
├── include
├── lib
│   └── python3.8
├── lib64 -> lib
├── pyvenv.cfg
├── share
│   └── python-wheels
└── 7 directories, 12 files
```

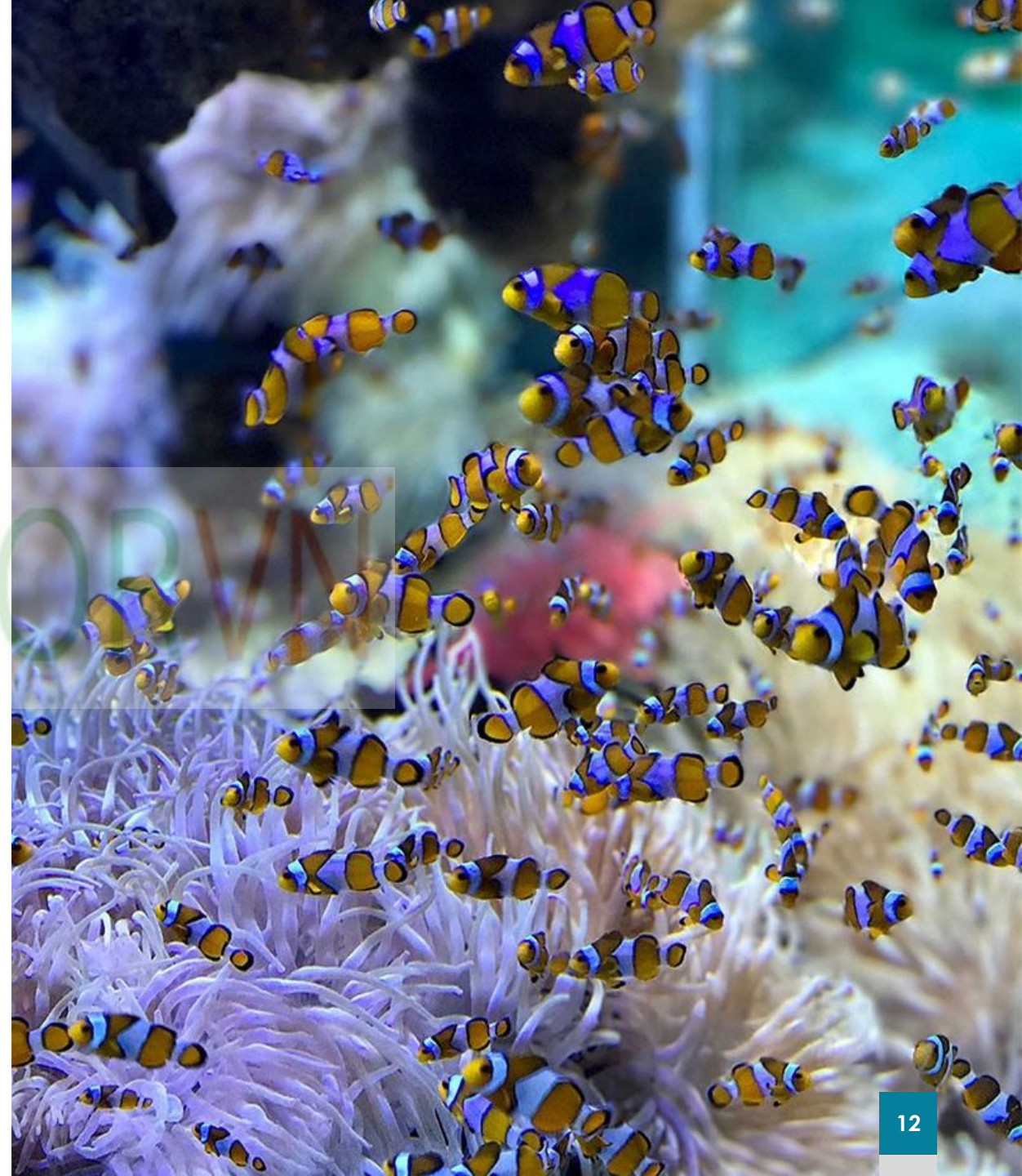
5. Basic scripting

Syntax

Python keywords

PEP8

How to execute Python script



The Zen of Python



1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.
7. Readability counts.
8. Special cases aren't special enough to break the rules.
9. Although practicality beats purity.
10. Errors should never pass silently.
11. Unless explicitly silenced.
12. In the face of ambiguity, refuse the temptation to guess.
13. There should be one-- and preferably only one --obvious way to do it.
14. Although that way may not be obvious at first unless you're Dutch.
15. Now is better than never.
16. Although never is often better than *right* now.
17. If the implementation is hard to explain, it's a bad idea.
18. If the implementation is easy to explain, it may be a good idea.
19. Namespaces are one honking great idea -- let's do more of those!

5.1. Syntax

- Cấu trúc 1 file Python cơ bản:

```
1  #!/usr/bin/env python3 } 1
2
3  # -*- coding: utf-8 -*-
4  # @Author: windows
5  # @Date: 2021-04-02 15:19:51
6  # @Last Modified by: windows
7  # @Last Modified time: 2021-04-03 10:09:57 } 2
8
9  def my_function():
10     .... my_name: str = "Pham Van A"
11     .... my_age: int = 20
12     .... print(f"My name is {my_name}, and my age is {my_age}") } 3
13
14 my_function()
```

- 1: python interpreter sẽ thực thi file mã python. Sử dụng `/usr/bin/env` để xác định interpreter.
- 2: (option) file header
- 3: python sử dụng **indent** để xác định các code thuộc về cùng 1 block chức năng. Ví dụ: các code thuộc về hàm `my_function` sẽ có cùng 1 cấp indent

```
help> keywords
```

Here is a list of the Python keywords. Enter any keyword to get more help.

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

5.2. Python keywords

"Keywords" là các từ được dành riêng và không được dùng để đặt tên trong lập trình Python

5.3. PEP

- PEP – Python Enhancement Proposal: guidelines và best practices để viết code Python. Được viết bởi Guido Van Rossum, Barry Warsaw, Nick Coghlan vào năm 2001 nhằm mục tiêu tăng tính “readability” và tính “nhất quán” trong việc lập trình Python. Docs: <https://www.python.org/dev/peps/pep-0008/>
- Naming: tên phải mang tính gợi nhớ, ý nghĩa rõ ràng.

Type	Rule	Example
function	Sử dụng chữ cái thường. Phân tách các từ bởi dấu gạch dưới	function_abc, make_decision
variable	Sử dụng chữ cái thường. Có thể sử dụng một kí tự, một từ hoặc nhiều từ. Phân tách các từ bởi dấu gạch dưới.	my_name, your_name
class	Chữ cái đầu của mỗi từ là in hoa. Không phân tách các từ với nhau bằng dấu gạch dưới. Cách viết này được gọi là camel case.	Animal, FlyingBird
constant	Sử dụng chữ cái IN HOA, một hoặc nhiều từ. Phân tách nhau bằng dấu gạch dưới.	CONFIG, USER_NAME

5.3. PEP

- Code layout:
 - Sử dụng 2 dòng trắng (blank line) để phân tách giữa 2 class
 - Sử dụng 1 dòng trắng để phân tách các hàm trong cùng 1 class, hoặc các hàm với nhau, hoặc phân tách logic 1 đoạn code.
 - Nên giới hạn độ dài 1 dòng code <80 ký tự.
- Indentation:
 - Sử dụng space, KHÔNG sử dụng tab.
 - Độ dài 4 space.
- Comment: bắt đầu bằng # và các comment nên cùng 1 indentation.
- Docstring: bắt đầu và kết thúc bằng 3 dấu nháy đôi `"""` `"""`

5.4. Execute Python script

1. Python shell:

- Thực thi từng dòng code và hiển thị kết quả, cũng như lỗi khi thực thi.

2. Executable file:

- File thực thi phải có quyền execute phù hợp với ownership của file đó.
- Python file phải có khai báo interpreter (shebang) rõ ràng, phù hợp.
- `./file.py`

3. Thực thi trực tiếp với Interpreter:

- Không yêu cầu file thực thi phải có quyền executable.
- Không yêu cầu file thực thi có khai báo interpreter.
- Nếu có khai báo interpreter shebang trong file thực thi, thì interpreter chỉ định sẽ overwrite.