**1. Bắt đầu**[**ℑ**](https://anandology.com/python-practice-book/getting-started.html#getting-started)

**1.1. Chạy trình thông dịch Python**[**ℑ**](https://anandology.com/python-practice-book/getting-started.html#running-python-interpreter)

Python đi kèm với một trình thông dịch tương tác. Khi bạn nhập pythonshell hoặc dấu nhắc lệnh, trình thông dịch python sẽ hoạt động với >>> dấu nhắc và chờ lệnh của bạn.

$ python

Python 3.7.4 (v3.7.4:e09359112e, Jul 8 2019, 14:54:52)

[Clang 6.0 (clang-600.0.57)] on darwin

Type "help", "copyright", "credits" or "license" for more information.

>>>

Bây giờ bạn có thể nhập bất kỳ biểu thức python hợp lệ nào tại dấu nhắc. python đọc biểu thức đã gõ, đánh giá nó và in kết quả.

**>>>** 42

42

**>>>** 4 + 2

6

**Vấn đề 1:** Mở trình thông dịch Python mới và sử dụng nó để tìm giá trị của .2 + 3

**1.2. Chạy tập lệnh Python**[**ℑ**](https://anandology.com/python-practice-book/getting-started.html#running-python-scripts)

Mở trình soạn thảo văn bản của bạn, nhập văn bản sau và lưu nó dưới dạng hello.py.

print("hello, world!")

Và chạy chương trình này bằng cách gọi . Đảm bảo bạn thay đổi thư mục nơi bạn đã lưu tệp trước khi thực hiện.python hello.py

$ python hello.py

hello, world!

**1.3. Loại dữ liệu**[**ℑ**](https://anandology.com/python-practice-book/getting-started.html#datatypes)

Python hỗ trợ tất cả các kiểu dữ liệu cơ bản và cũng có các kiểu dữ liệu phức hợp rất mạnh.

Python có số nguyên.

**>>>** 1 + 2

3

Python cũng khá giỏi trong việc xử lý các số rất lớn. Ví dụ: chúng ta hãy thử tính 2 lần tăng lũy ​​thừa 1000.

**>>>** 2 \*\* 1000

10715086071862673209484250490600018105614048117055336074437503883703510511249361224931983788156958581275946729175531468251871452856923140435984577574698574803934567774824230985421074605062371141877954182153046474983581941267398767559165543946077062914571196477686542167660429831652624386837205668069376

Đó là một con số khá lớn phải không? Bạn có đếm được nó có bao nhiêu chữ số không?

Python có số dấu phẩy động.

**>>>** 1.2 + 2.3

3.5

Python có chuỗi.

**>>>** "hello world"

'hello world'

**>>>** print("hello world")

hello world

Chuỗi có thể được đặt trong dấu ngoặc đơn hoặc dấu ngoặc kép. Cả hai đều giống hệt nhau. Trong Python, chuỗi rất linh hoạt và rất dễ làm việc với chúng.

**>>>** 'hello' + 'world'

'helloworld'

**>>>** "hello" \* 3

'hellohellohello'

**>>>** print("=" \* 40)

========================================

Hàm tích hợp lenđược sử dụng để tìm độ dài của chuỗi.

**>>>** len('helloworld')

10

Python cũng hỗ trợ chuỗi nhiều dòng. Chúng được đặt trong ba dấu ngoặc kép hoặc ba dấu ngoặc đơn.

text = """This is a multi-line string.

Line 2

Line 3

and the text may have "quotes" too.

"""

**>>>** print(text)

This is a multi-line string.

Line 2

Line 3

and the text may have "quotes" too.

Python hỗ trợ các mã thoát thông thường. \ncho biết dòng mới, \tcho biết một tab, v.v.

**>>>** print("a**\n**b**\n**c")

a

b

c

Python có danh sách. Danh sách là một trong những kiểu dữ liệu hữu ích nhất trong Python.

**>>>** x = ["a", "b", "c"]

**>>>** x

['a', 'b', 'c']

**>>>** len(x)

3

**>>>** x[1]

'b'

Python có một kiểu dữ liệu khác gọi là *tuple* để biểu diễn các bản ghi có chiều rộng cố định. Các bộ dữ liệu hoạt động giống như danh sách nhưng chúng không thể thay đổi được.

**>>>** point = (2, 3)

**>>>** point

(2, 3)

Khi viết các bộ dữ liệu, dấu ngoặc đơn có thể được bỏ qua trong hầu hết các trường hợp.

**>>>** point = 2, 3

**>>>** point

(2, 3)

Cũng có thể gán một bộ nhiều giá trị cùng một lúc:

**>>>** yellow = (255, 255, 0)

**>>>** r, g, b = yellow

**>>>** print(r, g, b)

255 255 0

Python có dictionarykiểu dữ liệu để biểu diễn các cặp tên-giá trị.

**>>>** person = {"name": "Alice", "email": "alice@example.com"}

**>>>** person['name']

'Alice'

**>>>** person['email']

'alice@example.com'

Python setcũng có kiểu dữ liệu. Tập hợp là một tập hợp các phần tử không có thứ tự.

**>>>** x = {1, 2, 3, 2, 1}

**>>>** x

{1, 2, 3}

Python có một booleanloại. Nó có hai giá trị đặc biệt Truevà Falseđại diện cho sự thật và sai lầm.

Cuối cùng, Python có một kiểu đặc biệt được gọi Nonelà không đại diện cho gì cả.

**>>>** x = **None**

**>>>** print(x)

None

Bây giờ bạn đã biết hầu hết các cấu trúc dữ liệu phổ biến của Python. Mặc dù chúng trông rất đơn giản nhưng để thành thạo chúng cần phải thực hành một chút. Đảm bảo bạn xem qua tất cả các ví dụ và bài tập thực hành trong các phần tiếp theo.