Python Basic

Ba Nguyễn

Syntax

Cú pháp Python rất đơn giản:

- Mỗi câu lệnh đặt trên một dòng
- Không cần kết thúc với dấu ;
- Phần thụt lề (indent) rất quan trọng

```
# Comment (Shortcut Ctrl + /)

print("Hello, what your name?")
name = input("> ")

print("Hello, " + name)
print("Nice to meet you 😍")
```

Variables

Biến là yếu tố cơ bản của bất kỳ ngôn ngữ lập trình nào. ***Biến là tên một vùng nhớ lưu trữ dữ liệu trong bộ nhớ máy tính.***

```
# Cú pháp khai báo biến trong Python:
my_name = "Ba"
my_age = 28
is_handsome = True
a_poem = """
Phận làm trai gõ phím bình thiên hạ
Chí anh hùng click chuột định giang sơn
"""
x = y = z = 1
x, y = 1, 2 # x = 1, y = 2
```

Variables

Quy tắc đặt tên biến:

- Tên biến *không bắt đầu bằng số*
- Tên biến *không chứa ký tự đặc biệt*
- Tên biến *không trùng với keyword trong Python*

Quy ước đặt tên biến (*PEP 8 - Python Enhancement Proposal*)

- Tên biến, hàm nên tuân thủ quy ước đặt tên *lower_case_with_underscore*
- Class hoặc Exception sử dụng *CapitalizeCase*

```
my_computer = "Dell"
my_phone_number = "096 533 828x"
```

Variables

Các kiểu dữ liệu cơ bản (*nguyên thủy - primitive*) trong Python bao gồm:

- *Số nguyên - **int***
- *Số thực - **float***
- *Logic - **bool***
- *Chuỗi - **str***
- *None*

```
type_int = 1
type_float = 0.5
type_bool = True  # False
type_str = "What's up bitches"

# Kiểm tra kiểu dữ liệu
type(type_int)
# <class 'int'>
```

Variables

Python là ngôn ngữ có kiểu động (*dynamic typing*) tương tự một số ngôn ngữ như JavaScript, biến được khai báo hoặc thay đổi giá trị mà không cần chỉ định kiểu dữ liệu, điều này khác biệt với một số ngôn ngữ có kiểu tĩnh (*static typing*) như C, Java

```java
// Java
int a = 0;
a = "Not amazing"; // ❌ Error
String b = false; // ❌ Error
```

```python
# Python
var = 0
var = "Amazing"  # 👌 Okey
var = True  # 👌 Okey
```

Type Annotation

Python 3 hỗ trợ cú pháp *ghi chú kiểu dữ liệu* (*Type Annotation*) cho biến.

Type Annotation giúp gợi ý viết code chặt chẽ hơn chứ không ràng buộc kiểu dữ liệu cho biến, tuy nhiên nên sử dụng Type Annotation giúp code *clear* hơn

```
# Type Annotation
a: int = 1
b: str = "Amazing"
```

💡 Ngoài biến, có thể ghi chú kiểu dữ liệu đầu vào của các tham số hàm, kiểu dữ liệu trả về từ hàm, class, ...

Mutable vs Immutable Types

Với các kiểu dữ liệu cơ bản, giá trị trong biến là *không thể thay đổi (immutable)*, khi cập nhật giá trị của biến, thực chất, biến sẽ được *gán cho một vùng nhớ khác lưu trữ giá trị mới*

```
# id(obj) cho biết địa chỉ vùng nhớ của biến
name = "Ba"
print(id(name))   # 1888929324400

name = "Tokuda"
print(id(name))   # 1888929324401
```

💡 Với các giá trị không còn được sử dụng, Python sẽ tự động dọn dẹp để giải phóng bộ nhớ

Mutable vs Immutable Types

Mutable thì ngược lại, các giá trị bên trong biến có thể thay đổi được

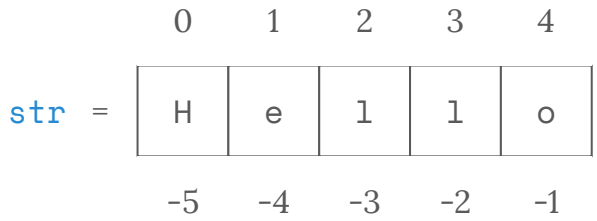
```
# Ví dụ với List
list = [1, 2, 3]
print(id(list))   # 140029576941504

list[1] = 0
print(id(list))   # 140029576941504
```

Strings

Chuỗi ***str*** là giá trị được đặt trong cặp dấu ‘ ’, “ ”, ‘‘‘ ’’’ hoặc “““ ”””, các ký tự trong chuỗi được đánh chỉ mục (*index*) tương ứng với thứ tự của nó - bắt đầu từ **0**. Python cho phép truy cập tới ký tự trong chuỗi thông qua chỉ mục

```
str = "Hello"
print(len(str))  # 5
print(str[0])  # H
print(str[-1])  # o
print(str[0:2])  # He
print(str[-4:-1])  # ell
print(str[:3])  # Hel
print(str[::2])  # Hlo
print(str[:])  # Hello
```




```
print(str[-2:10])  # ??
```

Escape Strings

Làm thế nào để in ra chuỗi: ***I'm Ba "đẹp trai"*** ??

```
print("I'm Ba "đẹp trai"") # ❌ Error

# Cú pháp Escape string
print("I'm Ba \"đẹp trai\"")
print('I\'m Ba "đẹp trai"')
```

💡 Một số ký tự đặc biệt thường dùng \', \", \\, \n

Formatted Strings

Python cho phép *nhúng giá trị của một biến, phương thức, hoặc một biểu thức, ...* vào một chuỗi với cú pháp ***f-strings***

```
name = "Ba"
age = 28
greeting = f"Chào, tớ là {name}, {age} tuổi!"
print(greeting)

# Cách cổ điển
greeting = "Chào, tớ là " + name + ", " + str(age) + " tuổi!"
print(greeting)
```

Strings Methods

Python cung cấp rất nhiều phương thức để thao tác với chuỗi, để sử dụng một phương thức, sử dụng cú pháp ***str.method()***

```
str = "Learn Python is fun"
print(str.lower())  # learn python is fun
print(str.upper())  # LEARN PYTHON IS FUN
print(str.capitalize())  # Learn python is fun
print(str.count("n"))  # 3
print(str.title())  # Learn Python Is Fun
print(str.find("on"))  # 10
print(str.swapcase())  # lEARN pYTHON IS FUN
print(str.split())  # ["Learn", "Python", "is", "fun"]
print(str.replace("fun", "very, very fun"))
# Learn Python is very, very fun
```

Numbers

Kiểu số, bao gồm số nguyên ***int***, số thực ***float*** và số phức ***complex***. Python cũng hỗ trợ định dạng số ***binary*** và ***hex*** với tiền tố (*prefix*) **0b** hoặc **0x**

```
num = 1  # int
num = 9.9  # float
num = 0b111  # int
print(num)  # int --> 7
print(bin(num))  # str --> 0b111
num = 0xffffff  # int
print(hex(num))  # str --> 16777215
num = 1 + 2j  # complex
print(num)  # (1 + 2j)
```

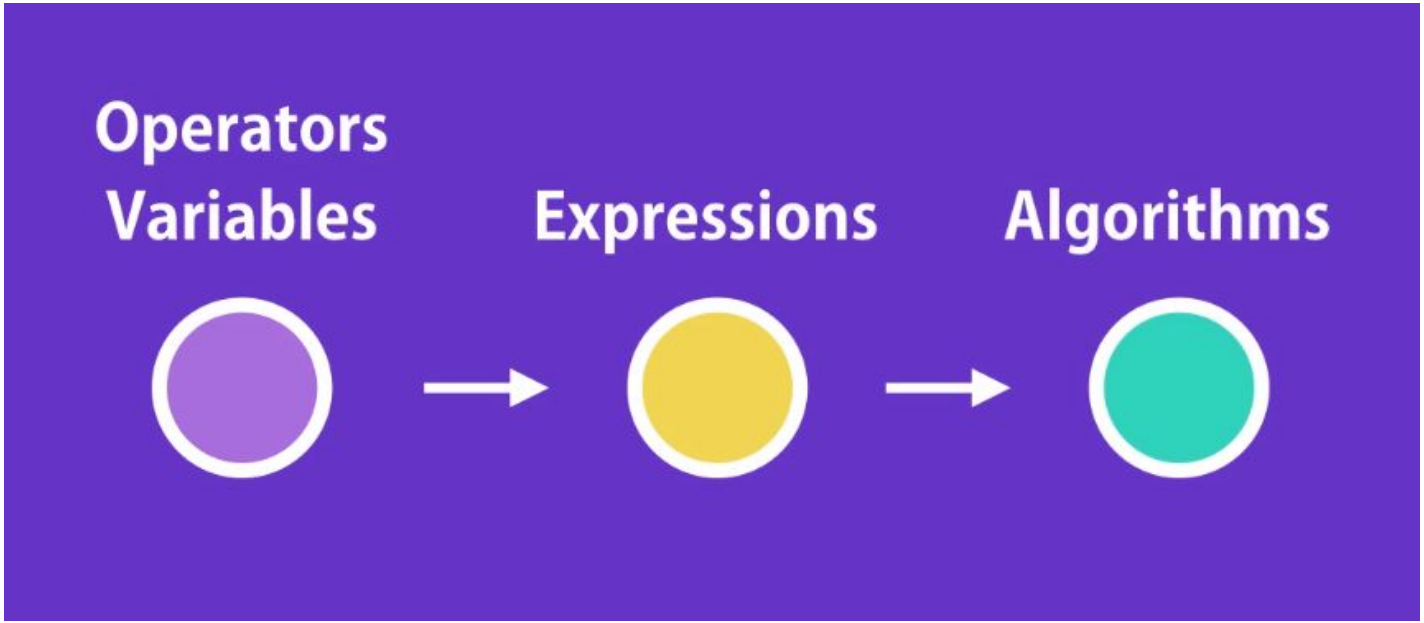
Working with Numbers

Khác với chuỗi, Python không cung cấp sẵn nhiều phương thức để làm việc với kiểu số. Thay vào đó, nó cung cấp một *module* ***math*** chứa các phương thức liên quan để xử lý số học

```
import math

print(round(35.425))  # 35
print(abs(-123))  # 123
print(math.pi)  # 3.14159265...
print(math.ceil(4.234545))  # 5
print(math.floor(35.825))  # 35
print(math.sqrt(4))  # 2.0
print(math.trunc(math.sqrt(4)))  # 2
print(math.pow(2, 3))  # 8.0
print(math.trunc(math.pow(2, 3)))  # 8
```

Operators



Basic Operators

Các loại toán tử cơ bản trong Python

```
x = 1 + 2  # 3
x = 10 - 5  # 5
x = 2 * 3  # 6
x = 4 / 2  # 2.0
x = 10 // 3  # 3
x = 10 % 3  # 1
x = 10 ** 2  # 100
```

```
x += 1
x -= 4
x *= 3
x /= x
x //= 4
x %= 6
x **= 4
```

```
a == b
a != b
a > b
a >= b
a < b
a <= b
```

💡 Trong Python không tồn tại toán tử ++, -- giống các ngôn ngữ khác

Type Conversion

Python không hỗ trợ chuyển đổi kiểu dữ liệu tự động (*strongly type language*) khi thực hiện tính toán các biểu thức giống một số ngôn ngữ khác như JavaScript (*weakly type language*), để chuyển đổi dữ liệu cần sử dụng các phương thức chuyển đổi kiểu rõ ràng

```
x = 10
print(x + "abc")          # ✗ Error

print(int(x))             # Falsy value
print(str(x))             # ""
print(float(x))           # 0
print(complex(x))         # [], {}
print(bool(x))            # None
                          # Các giá trị khác là True
```

Interaction

Python cung cấp phương thức ***input()*** cho phép nhập dữ liệu khi chương trình thực thi.

💡 Lưu ý dữ liệu nhận vào từ ***input()*** là kiểu ***str***

```
prompt = "> "

print("Chào, mình là Ba đẹp trai!")
print("Bạn tên gì?")
name = input(prompt)
print("Bạn nhiêu tuổi?")
age = input(prompt)

print(f"A mây zing! Gút chóp {name}")
```