Tìm hiểu Python

Thành viên:
Đinh Trọng Đạt
Nguyễn Anh Nhật Huy
Trương Quang Tuấn

Khai báo biến

- x = [giá trị]
- Không cần khai báo kiểu dữ liệu trước :
  - x = 5 //x có kiểu int
- Hoặc có thể khai báo kiểu dữ liệu cho biến:
  - x= int(5)

Numberic Type

- Int
  - x = 20
- Float
  - x=20.5
- Complex
  - x=1j

Text type

- Cách khai báo:
    - x = "Hello"
- String: nằm trong '...' hoặc "..." đều như nhau hoặc Multiline String: """...""" để khai báo 1 đoạn văn.
    - a = """Lorem ipsum dolor sit amet,consectetur adipiscing elit,sed do eiusmod tempor incididuntut labore et dolore magna aliqua."""

Text type

- String trong python là 1 mảng.
- Trong Python, không có kiểu dữ liệu ký tự (character), mà một ký tự là một chuỗi (string) có độ dài là 1 ký tự.
  - a = " CTK43"
    print(a[1]) //T

Boolean type

- Trả về True hoặc False

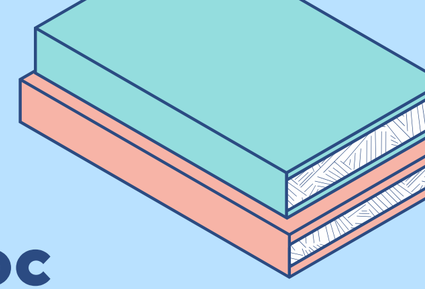
```
a = 200
b = 33

if b > a:
  print("B is greater than A")
else:
  print("B is not greater than A")
```

```
B is not greater than A
```

Cấu trúc điều khiển

Python hỗ trợ một số cấu trúc điều khiển thông dụng. Hầu hết các cấu trúc điều khiển đều dựa vào thụt đầu dòng (4 spaces) để tạo thành một block xử lý, thay vì sử dụng { ... } như các ngôn ngữ khác (PHP, Javascript).

1. Boolean và toán tử logic

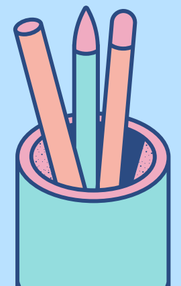


Toán tử số học

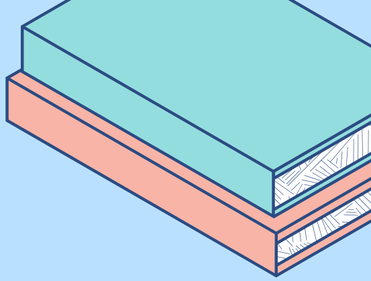
Toán tử	Mô tả
+	Toán tử cộng 2 giá trị
-	Toán tử trừ 2 giá trị
*	Toán tử nhân 2 giá trị
/	Toán tử chia 2 giá trị (chia ra số thập phân)
%	Toán tử chia 2 giá trị lấy phần dư
//	Toán tử chia 2 giá trị, làm tròn xuống
**	Toán tử mũ

Toán tử số học

Toán tử	Mô tả
==	So sánh giá trị của các đối số xem có bằng nhau hay không
!=	So sánh giá trị của các đối số xem có khác nhau hay không.
<	Dấu < đại diện cho phép toán nhỏ hơn
>	Dấu > đại diện cho phép toán lớn hơn
>=	Dấu > đại diện cho phép toán nhỏ hơn hoặc bằng
<=	Dấu > đại diện cho phép toán lớn hơn hoặc bằng



1. Boolean và toán tử logic

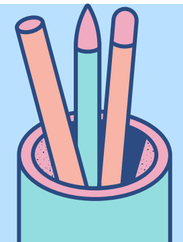


Toán tử gán

Toán tử	Mô tả
=	Toán tử gán cho 1 biến khác
+=	Toán tử này cộng rồi gán lại cho biến đó
-=	Toán tử này trừ rồi gán lại cho biến đó
*=	Toán tử này nhân rồi gán lại cho biến đó
/=	Toán tử này chia rồi gán lại cho biến đó
%=	Toán tử này chia lấy dư rồi gán lại cho biến đó
**=	Toán tử này tính mũ rồi gán lại cho biến đó
//=	Toán tử này chia làm tròn xuống rồi gán lại cho biến đó

Toán tử logic

Toán tử	Mô tả
and	Giống toán tử &&, đúng nếu 2 vế của end là True, còn lại là False
or	Giống toán tử ??, đúng nếu 1 trong 2 vế của or là True, còn lại là False
not	Giống toán tử !, dang phủ định
in	Nếu 1 đối số thuộc 1 tập nào đó thì trả về là True, ngược lại là False
not in	Ngược lại của in
is	Toán tử này trả về nếu 2 vế của toán tử: a == b
not is	Ngược lại của is

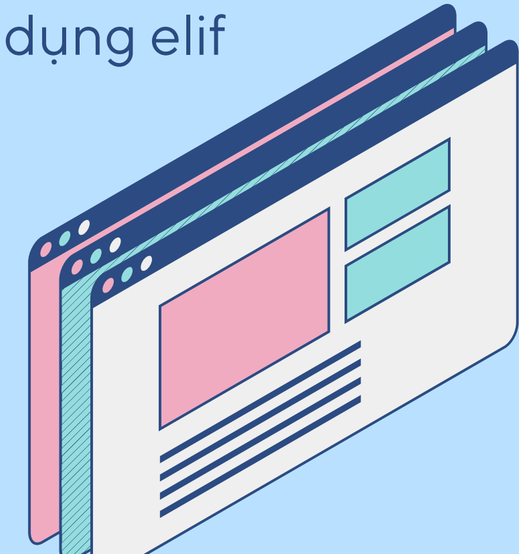


2. Câu lệnh if

- Câu lệnh if được sử dụng để kiểm tra một điều kiện: nếu điều kiện là đúng sẽ chạy một khối các câu lệnh (được gọi là if-block).
- **Cú pháp :**

```
if condition:
    statements
```

- Khối else sẽ được thực thi nếu điều kiện trong if là sai
- Khi ta phải kiểm tra thêm một điều kiện nữa nếu trong if là sai ta có thể sử dụng elif
- Chúng ta có thể đặt các khối lệnh if lồng nhau:



2. Cấu trúc vòng lặp

- Vòng lặp for

- Vòng lặp for ở trong Python có tác dụng lặp các biến dữ liệu có kiểu dữ liệu list, string,...
- **Cú pháp :**

```
for variable in data:
    # code
```

- Câu lệnh for của Python lặp qua các phần tử của 1 danh sách truyền vào ~ tương tự với foreach - lặp tất cả các phần tử.

```
words = ['cat', 'dog', 'bird'];
for w in words:
    print(w)

# cat
# dog
# bird
```

- Vòng lặp while

- Vòng lặp while trong python tương tự trong các ngôn ngữ khác, lặp khi thỏa mãn điều kiện cho trước.
- **Cú pháp :**

```
while <condition>:
    # code
```

- break và continue: Giống với PHP, break cho phép thoát khỏi vòng lặp, còn continue cho phép bỏ qua lượt chạy hiện tại của vòng lặp và chạy tiếp.

```
i = 0
while (i <= 10):
    print(i)
    if i == 5:
        break
    i += 1

for i in ‘i am a robot’:
    if i == ‘ ’:
        continue
    print(i, end=’’)
```

3. Exception

- Lỗi xảy ra trong quá trình thực thi một chương trình, khi thực hiện 1 đoạn code nào đó mà có thể xảy ra lỗi trong quá trình chạy mà ta chưa thể xác định được trước, nó có thể làm cho chương trình bị chết, chết trang, ... hoặc khi lưu vào db có thể 1 lỗi nào đó xảy ra khiến dữ liệu lưu vào bị gián đoạn tạo ra các dữ liệu lỗi.

- **Cú pháp :**

```
try:
// code
except:
// ngoai le
finally:
// run every time
```

Hàm

- Cách khai báo: def function_name();
- Có thể truyền thông tin vào hàm qua đối số:
  - def function_name(fname);

```
def my_function(fname):
  print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")
my_function("Linus")
```

```
Emil Refsnes
Tobias Refsnes
Linus Refsnes
```

Lambda (hàm ẩn danh)

- Cách khai báo: lambda agrument : expression

```
double = lambda x: x * 2

print(double(5))

# 10
```

VÍ DỤ