TÁC GIẢ: 8 SYNC

CỘNG ĐỒNG

- Kevin Nguyễn
- Nhóm Chia Sẻ Công Nghệ
- Nhóm BlockChain
- Tiktok: 8 Sync
- Youtube: 8 Sync Dev
- Zalo

KHÓA HỌC:

- Fullstack Python
- Fullstack Nextjs
- Fullstack Android-IOS

Tài liệu sẽ được cập nhật định kì và thông báo trong group nên các bạn chú ý nhen .

BÀI 20. PYTHON - TOÁN TỬ BITWISE

Toán tử bitwise trong Python thường được sử dụng để thực hiện các phép toán bitwise trên các đối tượng kiểu số nguyên. Tuy nhiên, thay vì xem xét đối tượng như một thể, nó được xem xét như một chuỗi các bit. Các phép toán khác nhau được thực hiện trên từng bit trong chuỗi.

Python có sáu toán tử bitwise - &, |, ^, ~, << và >>. Tất cả các toán tử này (ngoại trừ ~) đều là nhị phân, có nghĩa là chúng hoạt động trên hai toán hạng. Mỗi toán hạng là một chữ số nhị phân (bit) 1 hoặc 0.

Dưới đây là các toán tử bitwise trong Python:

- Toán tử AND bitwise
- Toán tử OR bitwise
- Toán tử XOR bitwise
- Toán tử NOT bitwise
- Toán tử Dịch Trái bitwise
- Toán tử Dịch Phải bitwise

PYTHON TOÁN TỬ AND BITWISE (&)

Toán tử AND bitwise có một số tương đồng với toán tử and logic. Nó chỉ trả về True nếu cả hai toán hạng bit đều là 1 (tức là True). Tất cả các kết hợp là −

- 0 & 0 là 0
- 1 & 0 là 0
- 0 & 1 là 0
- 1 & 1 là 1

Ví dụ về Toán Tử AND Bitwise trong Python:

< 8 Sync Dev />

```
a = 60
b = 13
print ("a:", a, "b:", b, "a & b:", a & b)
```

Kết quả sẽ là:

< 8 Sync Dev />

```
a: 60 b: 13 a & b: 12
```

Để hiểu cách Python thực hiện phép toán, lấy giá trị nhị phân tương ứng của mỗi biến.

< 8 Sync Dev />

```
print ("a:", bin(a))
print ("b:", bin(b))
```

Sẽ tạo ra đầu ra sau:

< 8 Sync Dev />

```
a: 0b111100
b: 0b1101
```

PYTHON TOÁN TỬ OR BITWISE (|)

Ký hiệu "|" (gọi là dấu pipe) là toán tử OR bitwise. Nếu bất kỳ bit toán hạng nào là 1, kết quả là 1, nếu không, là 0.

- 0 | 0 là 0
- 0 | 1 là 1
- 1 | 0 là 1
- 1 | 1 là 1

Ví dụ về Toán Tử OR Bitwise trong Python:

< 8 Sync Dev />

```
a = 60
b = 13
print ("a:", a, "b:", b, "a | b:", a | b)
```

Kết quả sẽ là:

< 8 Sync Dev />

```
a: 60 b: 13 a | b: 61
```

PYTHON TOÁN TỬ XOR BITWISE (^)

Thuật ngữ XOR đại diện cho OR độc quyền. Nó có nghĩa là kết quả của phép toán OR trên hai bit sẽ là 1 nếu chỉ có một trong hai bit là 1.

- 0 ^ 0 là 0
- 0 ^ 1 là 1
- 1 ^ 0 là 1
- 1 ^ 1 là 0

Ví dụ về Toán Tử XOR Bitwise trong Python:

< 8 Sync Dev />

```
a = 60
b = 13
print ("a:", a, "b:", b, "a ^ b:", a ^ b)
```

Kết quả sẽ là:

< 8 Sync Dev />

```
a: 60 b: 13 a ^ b: 49
```

Chúng ta hiện thực phép XOR bitwise theo cách thủ công.

< 8 Sync Dev />

```
0011 1100
    ^
0000 1101
-------------
0011 0001
```

Hàm int() sẽ cho kết quả 00110001 là 49.

< 8 Sync Dev />

```
int('00110001',2)
```

PYTHON TOÁN TỬ NOT BITWISE (~)

Toán tử này là tương đương nhị phân của toán tử NOT logic. Nó đảo ngược từng bit để 1 được thay thế bằng 0 và ngược lại, và trả về bù của số ban đầu. Python sử dụng phương pháp của 2. Đối với số nguyên dương, nó được thu được đơn giản bằng cách đảo ngược các bit. Đối với số âm, -x, nó được viết bằng biểu diễn bit cho (x-1) với tất cả các bit đảo ngược (chuyển từ 1 thành 0 hoặc từ 0 thành 1).

Ví dụ về Toán Tử NOT Bitwise trong Python:

< 8 Sync Dev />

```
a = 60
print ("a:", a, "~a:", ~a)
```

Kết quả sẽ là:

< 8 Sync Dev />

```
a: 60 ~a: -61
```

PYTHON TOÁN TỬ DỊCH TRÁI BITWISE (<<)

Toán tử dịch trái dịch các bit có ý nghĩa nhất sang phải bởi số nằm bên phải của ký hiệu "<<" . Vì vậy, "x << 2" làm cho hai bit của biểu diễn nhị phân của x bị dịch sang phải.

Ví dụ về Toán Tử Dịch Trái Bitwise trong Python:

< 8 Sync Dev />

```
a = 60
print ("a:", a, "a << 2:", a << 2)
```

Kết quả sẽ là:

< 8 Sync Dev />

```
a: 60 a << 2: 240
```

PYTHON TOÁN TỬ DỊCH PHẢI BITWISE (>>)

Toán tử dịch phải dịch các bit có ý nghĩa nhỏ nhất sang trái bởi số nằm bên phải của ký hiệ ">>". Do đó, "x >> 2" làm cho hai bit của biểu diễn nhị phân của x bị dịch sang trái.

Ví dụ về Toán Tử Dịch Phải Bitwise trong Python:

```
< 8 Sync Dev />
```

```
a = 60
print ("a:", a, "a >> 2:", a >> 2)
```

Kết quả sẽ là:

```
< 8 Sync Dev />
```

```
a: 60 a >> 2: 15
```

Hoạt động dịch phải thủ công trên 60 được thể hiện dưới đây:

```
< 8 Sync Dev />
```

```
0011 1100
>>
2
------------
0000 1111
```

Sử dụng hàm int() để chuyển đổi số nhị phân trên thành số nguyên. Đó là 15.

```
< 8 Sync Dev />
```

```
int('00001111',2)
```

Đây là hướng dẫn về các toán tử bitwise trong Python. Đây là một phần quan trọng của ngôn ngữ lập trình Python, đặc biệt hữu ích khi làm việc với các phép toán bitwise trên các số nguyên.