TIN HỌC CƠ SỞ
Phần 2: Ngôn ngữ lập trình Python

CHƯƠNG 7.
TOÁN TỬ CƠ BẢN TRONG PYTHON

Tin cơ sở (LT): 010100229802 - DHKL16A1HN, - DHKL16A2HN

GV. Cao Diệp Thắng

Mục tiêu chương

Nắm vững:

- ❑ Khái niệm toán tử.
- ❑ Các toán tử cơ bản trong Python
  1. Toán tử số học - Arithmetic Operators.
  2. Toán tử so sánh - Comparison (Relational) Operators
  3. Toán tử gán - Assignment Operators.
  4. Toán tử bitwise - Bitwise Operators
  5. Toán tử logic - Logical Operators.
  6. Toán tử thành phần - Membership Operators.
  7. Toán tử nhận dạng - Indentity Operators.

Nội dung

1. Toán tử số học- Arithmetic Operators.
2. Toán tử so sánh-Comparison (Relational) Operators
3. Toán tử gán - Assignment Operators.
4. Toán tử bitwise - Bitwise Operators
5. Toán tử logic - Logical Operators.
6. Toán tử thành phần - Membership Operators.
7. Toán tử nhận dạng - Indentity Operators.

7.1. TOÁN TỬ

Trong python, các toán tử được được cũng cấp trong module Operator (Lib/operator.py) - Là một bộ các hàm chức năng, tương ứng với các toán tử của Python.

- ❑ Toán tử được sử dụng để thao tác với giá trị và các biến.
- ❑ Toán tử có thể làm việc với các đối tượng riêng biệt và trả về kết quả.
- ❑ Dữ liệu được gọi là toán hạng hoặc đối số.
- ❑ Toán tử được đại diện bởi các từ khóa hoặc các ký tự đặc biệt.

Ví dụ: đối với các toán tử định danh, chúng ta sử dụng từ khóa "**is**" và "**is not**".

7.2. TOÁN TỬ SỐ HỌC (ARITHMETIC OPERATORS)

- Toán tử số học thực hiện các phép tính số học khác nhau như cộng, trừ, nhân, chia, tìm phần dư, số mũ, v.v.
- Có nhiều phương pháp khác nhau để tính toán số học trong Python như sử dụng hàm **eval**, khai báo biến & tính toán hoặc gọi hàm.

Bảng 7.1. Toán tử số học trong Python

Toán tử	Mô tả	Ví dụ (a=5; b=3)
+	Cộng a+b	8
-	Trừ a-b	2
*	Nhân a*b	15
/	Chia a/b	1.6666666666666667
%	Chia dư a%b	2
**	a mũ b	125
//	Thực hiện phép chia, làm tròn cận dưới (Floor Division) Vd: 0,57 => 0, 0.9 => 0. -07 => -1 -0.1 => -1	1

7.3. TOÁN TỬ SO SÁNH (COMPARISON OPERATORS)

❑ Toán tử so sánh còn được gọi là toán tử quan hệ.
Các toán tử so sánh khác là (== ,! =, <>,>, <=,...)

Bảng 7.2. Toán tử so sánh trong Python

Toán tử	Mô tả	Ví dụ (a=1,b=2)
==	So sánh bằng	False
!=	So sánh khác	True
>	So sánh lớn hơn	False
<	So sánh nhỏ hơn	True
>=	So sánh lớn hơn hoặc bằng	False
<=	So sánh nhỏ hơn hoặc bằng	True

7.4. TOÁN TỬ GÁN (ASSIGNMENT OPERATORS)

- Toán tử gán trong Python được sử dụng để gán giá trị của toán hạng bên phải cho toán hạng bên trái.
- Các toán tử gán khác nhau được sử dụng trong Python là (+=, -=, *=, /=, ...).

Bảng 7.3. Toán tử gán trong Python

Toán tử	Mô tả	Ví dụ (a=5; c=2)
=	Gán toán hạng thứ hai cho toán hạng thứ nhất	c = a
+=	Cộng toán hạng sau vào toán hạng đầu và gán kết quả cho toán hạng đầu	c += a ↔ c = c + a
-=	Trừ toán hạng sau khỏi toán hạng đầu và gán kết quả cho toán hạng đầu	c -= a ↔ c = c - a
*=	Nhân toán hạng sau vào toán hạng đầu và gán kết quả cho toán hạng đầu	c *= a ↔ c = c * a
/=	Chia toán hạng sau vào toán hạng đầu và gán kết quả cho toán hạng đầu	c /= a ↔ c = c / a
%=	Chia và lấy phần dư của toán hạng sau vào toán hạng đầu và gán kết quả là toán hạng đầu	c %= a ↔ c = c % a
**=	Thực hiện phép tính số mũ và gán kết quả cho toán hạng đầu	c **= a ↔ c = c ** a
//=	Thực phép chia làm tròn cận dưới và gán kết quả cho toán hạng đầu	c //= a ↔ c = c // a

Ví dụ 7.1.

```
>>> a=7
>>> c=5
>>> b=9
>>> a=b
>>> print(a)
9
>>> c+=b
>>> print(c)
14
>>> c-=b
>>> print(c)
5
>>> c/=b
>>> print(c)
0.5555555555555556
```

```
>>> c*=b
>>> print(c)
5.0
>>> c%=b
>>> print(c)
5.0
>>> c**=b
>>> print(c)
1953125.0
>>> c//=b
>>> print(c)
217013.0
```

7.5. TOÁN TỬ BITWISE (BITWISE OPERATORS)

- Các toán tử thao tác bit hoạt động trên các bits và thực hiện các phép toán với độ chi tiết trên từng bit.

Bảng 7.4. Các thức hoạt động toán tử Bitwwise trong python

Toán tử	Mô tả	Cú pháp	Ví dụ: a=0011 1100, b=0000 1101
&	Phép **AND** trên các bits	**a & b**	(a&b) là 0000 1100
|	Phép **OR** trên các bits	**a | b**	(a | b) = 61 (nghĩa là 0011 1101)
^	Phép **XOR** trên các bits (đặt mỗi bit thành 1 nếu chỉ một trong hai bit là 1).	**a ^ b**	(a ^ b) = 49 (nghĩa 0011 0001)
~	Đảo bit/Phép **NOT** trên các bit	**~a**	(~a) =-61
<<	Phép dịch trái một bit	**a << = b**	a<<2=240 là 1101 0000
>>	Phép dịch phải một bit	**a >> = b**	a>>2=15 (nghĩa là 0000 1111)

❑ Ứng dụng toán tử bitwise trong bài toán tập hợp:

Ví dụ trường hợp đặt hằng số đại diện 12 con giáp này trong lập trình

Thông thường mọi người hay đặt: (**Tý**: **1**, **Sửu**: **2**, **Dần**: **3**, **Mão**: **4**, …, **Hợi**:**12**)

Để ứng dụng phép **bitwise** chúng ta đặt :

(**Tý**: 1<<0, **Sửu**: 1<<1, **Dần**: 1<<2, **Mão**: 1<<3, …, **Hợi**: 1<<11)

❑ khi đặt như vậy để xác định các tập hợp trên không hề giao nhau và đều khác 0 (tập rỗng ϕ)

a. Phép hợp (Union): Khi thực hiện phép **a | b**, ta có thể thu được biểu diễn bit của kết quả phép **A ∪ B**. ($A \cup B$ <=> $a \mid b$)

Ví dụ :

Tập hợp A = {**Tý**, **Sửu**, **Dần** } (giá trị binary a là **111**)
Tập hợp B = {**Tý**, **Sửu**, **Mão**} (giá trị binary b là **1011**)
–> **A ∪ B = {**Tý, Sửu, Dần, Mão**}** (tương ứng với giá trị binary a|b là **1111**)

b. Phép giao (Intersection): Khi thực hiện phép **a & b**, ta có thể thu được biểu diễn bit của kết quả phép **A ∩ B**. (*A ∩ B <=>a & b)*

Ví dụ:

A = {Tý, Sửu, Dần} (giá trị binary a là 111)

B = {Tý, Sửu, Mão} (giá trị binary b là 1011)

→ **A ∩ B= {Tý, Sửu}** (tương ứng với giá trị binary a & b là **11**)

Phép thuộc: (trường hợp con của phép giao)

Ví dụ:

A = {Tý, Sửu, Dần} (giá trị binary a là 111)

x = {Sửu} (giá trị binary b là 10)

→**x ∩ A = {Sửu}** (tương ứng với giá trị binary x & a là **10**)

c. Phép Hiệu (Difference): Khi thực hiện phép **a ^ (a&b)**, ta có thể thu được biểu diễn bit của kết quả phép A \ B (**A \ B <=> a ^ (a & b)**)

Ví dụ

Tập hợp A = {**Tý**, **Sửu**, **Dần** } (giá trị binary a là **111**)

Tập hợp B = {**Tý**, **Sửu**, **Mão**} (giá trị binary b là **1011**)

→ **A \ B** = {Dần} (tương ứng với giá trị binary **a^(a&b)** là **100**)

7.6. TOÁN TỬ LOGIC (LOGICAL OPERATORS)

- Toán tử logic trong Python được sử dụng cho các câu lệnh điều kiện là đúng hoặc sai.
- Toán tử logic trong Python bao gồm AND, OR và NOT.

Bảng 7.5. Toán tử logic AND, OR,N NOT

Toán tử	Mô tả	Ví dụ
And	Phép toán luận lý VÀ (AND) trên 2 giá trị. Kết quả là True nếu cả 2 giá trị đều là True	>>> **print**(2 < 10 and 2 < 40) #> True

or	Phép toán luận lý HOẶC(OR) trên 2 giá trị.
	Kết quả là True nếu 1 trong 2 giá trị là True
		>>> **print**(2<10 and 2 >40)
		False
		>>> **print**(2<10 or 2 > 40)

not	Phủ định True thành False và ngược lại	>>> **print**(not 2 > 10)
		True
		>>> **print**(not 2 < 10)
		False

7.7. TOÁN TỬ THÀNH PHẦN (MEMBERSHIP OPERATORS)

- Toán tử này thường được dùng để kiểm tra xem 1 đối số có nằm trong 1 tập hợp đối số hay không (**list, set, tuple**)

Bảng 7.6. Toán tử thành phần in, not in

Toán tử	**Mô tả**	**Ví dụ (a=10, b=20)** **List=[1,2,3,4,5]**
in	Trả về True nếu tìm thấy giá trị trong danh sách cụ thể, ngược lại trả về False.	>>>a in list False
not in	Trả về true nếu không tìm thấy giá trị trong danh sách cụ thể, ngược lại trả về False.	>>>b not in list True

7.8. TOÁN TỬ ĐỊNH DANH(IDENTITY OPERATORS)

- Mọi thứ trong Python đều là **một đối tượng** và mỗi đối tượng được lưu trữ tại một vị trí bộ nhớ cụ thể nào đó.
- Để so sánh vị trí bộ nhớ của hai đối tượng, toán tử định danh được sử dụng.
- Hai toán tử định danh được sử dụng trong Python là (**is**, **is not**).
- Các toán tử **is** và **is not** kiểm tra xem hai biến có tham chiếu đến cùng một đối tượng trong bộ nhớ hay không.

is trả về : **đúng (True)** nếu hai trỏ tới một đối tượng và trả về **sai (False)** trong trường hợp ngược lại.

is not trả về : **sai (False)** nếu hai biến cùng trỏ tới một đối tượng và trả về **đúng (True)** trong trường hợp ngược lại.

Bảng 7.7. Toán tử định danh

Toán tử	Mô tả	Ví dụ (a=10,b=10)

is	Trả về True nếu các biến ở cả hai bên toán tử cùng trỏ đến cùng một đối tượng, ngược lại trả về False	>>> a==b
		True
		>>> a **is** b
		True

is not	Trả về False nếu các biến ở cả hai bên toán tử cùng trỏ đến cùng một đối tượng, ngược lại trả về False	>>>id(a)==id(b) True >>>a **is not** b False

Ghi chú: *Hàm **id(object)** là hàm tích hợp sẵn trong Python trả về giá trị là một số nguyên duy nhất dùng để định danh đối tượng.*

7.9. ĐỘ ƯU TIÊN TOÁN TỬ

Operator	Description
**	Số mũ - Exponentiation (raise to the power)
~ + -	Complement, unary plus and minus (method names for the last two are +@ and -@)
* / % //	Nhân, chia, tính phần nguyên, tính số dư (Multiply, divide, modulo and floor division)
+ -	Phép cộng, trừ (Addition and subtraction)
>> <<	Toán tử dịch bit phải trái (Right and left bitwise shift)
&	Bitwise 'AND'td>
^ |	Bitwise exclusive `OR' and regular `OR'
<= < > >=	Toán tử so sánh (Comparison operators)
<> == !=	Toán tử ngang bằng (Equality operators)
= %= /= //= -= += *= **=	Toán tử gán (Assignment operators)
is is not	Toán tử định danh (Identity operators)
in not in	Toán tử thành viên (Membership operators)
not or and	Toán tử logic (Logical operators)

Các toán tử nằm trong cùng một ô sẽ có được tính từ trái sang phải.

7.10. MỘT SỐ VÍ DỤ

Ví dụ 7.3

```
>>> salary = 1200
>>> cond = salary >=5000 and salary <=7000
>>> print(cond)
False
>>> salary = 7000
>>> cond = salary >=5000 and salary <=7000
>>> print(cond) # In ra True
True
```

```
salary = 1200
cond = 5000 <= salary <=7000
print(cond) # In ra False

salary = 7000
cond = 5000 <= salary <=7000
print(cond) # In ra True
```

```
depid = 70
cond = depid==50 or depid==70 or depid==90
print(cond) # in ra True
```

```
depid = 70
cond = depid in [50,70,90]
print(cond) # in ra True
```

Ví dụ 7.4

```
1    #Tìm nhân viên có lương >=5000 và ở các phòng 50 hoặc 60
2    luong = 7000
3    phong = 50
4    cond = luong>=5000 and (phong==60 or phong==50)
5    print(cond) #In ra True
```

```
1    #Tìm nhân viên có lương >=5000 và ở các phòng 50 hoặc 60
2    luong = 2000
3    phong = 50
4    cond = luong>=5000 and (phong==60 or phong==50)
5    print(cond) #In ra False
```

Câu hỏi củng cố bài

1. Khẳng định nào sau đây về Python là đúng?

A. Python là một ngôn ngữ lập trình cấp cao.

B. Python là một ngôn ngữ thông dịch.

C. Python là ngôn ngữ lập trình hướng đối tượng.

D. Tất cả các đáp án đều đúng.

Câu hỏi củng cố bài

2. Biểu thức [(x+y)*z]-(x2-y2) chuyển sang Python là:

A. ((x+y)*z)-(x2-y2)

B. ((x+y)*z)-(x*x-y*y)

C. ((x+y)*z)-(x2-y2)

D. (x+y)*z-x*x-y*y

Câu hỏi củng cố bài

3. Trong phép toán số học với số nguyên, phép toán lấy phần dư trong Python là:

A. %

B. mod

C. //

D. div

Câu hỏi củng cố bài

4. Trong phép toán số học với số nguyên, phép toán lấy phần nguyên trong Python là:

A. %

B. mod

C. //

D. div

Câu hỏi củng cố bài

5. x^2 được biểu diễn trong Python là:

A. x**2

B. x*2

C. x2

D.x**

Câu hỏi củng cố bài

6. Trong Python khi viết **x+=5** có nghĩa là:

A. Giảm x đi 5 đơn vị

B. Tăng x lên 5 đơn vị

C. Tăng x lên 1 đơn vị

D. X giữ nguyên giá trị

Câu hỏi củng cố bài

7. Đoạn code dưới đây trả về kết quả nào ?

```
x = 3
x >>= 6
print(x)
```

A. 1 **B**. 3 **C**. 2 **D**. 0

Câu hỏi thảo luận

1. Các loại dữ liệu được hỗ trợ trong Python là gì?
2. Trình bày cách chuyển đổi một số nguyên thành một ký tự unicode trong python?
3. Trình bày các loại toán tử trong python?
4. Nêu các toán tử số học trong python? Ví dụ?
5. Nêu thứ tự ưu tiên các toán tử quan hệ? Ví dụ?
6. Ý nghĩa toán tử membership trong python?

Bài tập vận dụng

Các bài tập trong TLHT chương 7.