Python - Operadores

Operadores Python

Os operadores Python são símbolos especiais (às vezes chamados de palavras-chave) usados para executar certas operações mais comumente necessárias em um ou mais operandos (valor, variáveis ou expressões).

Tipos de operadores em Python

A linguagem Python suporta os seguintes tipos de operadores -

- Operadores aritméticos
- Operadores de comparação (relacionais)
- Operadores de Atribuição
- Operadores lógicos
- Operadores bit a bit
- Operadores de adesão
- Operadores de identidade

Vamos dar uma olhada em todos os operadores, um por um.

Operadores Aritméticos Python

Operadores aritméticos são usados para realizar operações matemáticas básicas, como adição, subtração, multiplicação, etc.

Suponha que a variável a contenha 10 e a variável b contenha 20, então

Operador	Nome	Exemplo
+	Adição	uma + b = 30
-	Subtração	uma – b = -10
*	Multiplicação	uma * b = 200

/	Divisão	b / a = 2
%	Módulo	b % a = 0
**	Expoente	a**b =10**20
//	Divisão de Piso	9//2 = 4

Exemplo de operadores aritméticos Python

```
a = 21
b = 10
c = 0

c = a + b
print ("a: {} b: {} a+b: {}".format(a,b,c))

c = a - b
print ("a: {} b: {} a-b: {}".format(a,b,c) )

c = a * b
print ("a: {} b: {} a*b: {}".format(a,b,c))

c = a / b
print ("a: {} b: {} a/b: {}".format(a,b,c))

c = a % b
print ("a: {} b: {} a%b: {}".format(a,b,c))

a = 2
b = 3
c = a**b
print ("a: {} b: {} a**b: {}".format(a,b,c))

a = 10
b = 5
c = a//b
print ("a: {} b: {} a//b: {}".format(a,b,c))
```

Saída

```
a: 21 b: 10 a+b: 31
a: 21 b: 10 a-b: 11
a: 21 b: 10 a*b: 210
a: 21 b: 10 a/b: 2.1
a: 21 b: 10 a%b: 1
a: 2 b: 3 a**b: 8
a: 10 b: 5 a//b: 2
```

Operadores de comparação Python

Os operadores de comparação comparam os valores de cada lado deles e decidem a relação entre eles. Eles também são chamados de operadores relacionais.

Suponha que a variável **a** contenha 10 e a variável **b** contenha 20, então

Operador	Nome	Exemplo
==	Igual	(a == b) não é verdade.
!=	Não igual	(uma! = b) é verdadeiro.
>	Maior que	(a > b) não é verdade.
<	Menor que	(a < b) é verdadeiro.
>=	Melhor que ou igual a	(a >= b) não é verdade.
<=	Menos que ou igual a	(a <= b) é verdadeiro.

Exemplo de operadores de comparação Python

```
a = 21
b = 10
if ( a == b ):
   print ("Line 1 - a is equal to b")
else:
   print ("Line 1 - a is not equal to b")

if ( a != b ):
   print ("Line 2 - a is not equal to b")
else:
   print ("Line 2 - a is equal to b")
```

```
if ( a < b ):
   print ("Line 3 - a is less than b" )
else:
   print ("Line 3 - a is not less than b")

if ( a > b ):
   print ("Line 4 - a is greater than b")
else:
   print ("Line 4 - a is not greater than b")

a,b=b,a #values of a and b swapped. a becomes 10, b becomes 21

if ( a <= b ):
   print ("Line 5 - a is either less than or equal to  b")
else:
   print ("Line 5 - a is neither less than nor equal to  b")

if ( b >= a ):
   print ("Line 6 - b is either greater than  or equal to b")
else:
   print ("Line 6 - b is neither greater than  nor equal to b")
```

Saída

```
Line 1 - a is not equal to b
Line 2 - a is not equal to b
Line 3 - a is not less than b
Line 4 - a is greater than b
Line 5 - a is either less than or equal to  b
Line 6 - b is either greater than  or equal to b
```

Operadores de Atribuição Python

Operadores de atribuição são usados para atribuir valores a variáveis. A seguir está uma tabela que mostra todos os operadores de atribuição Python.

Operador	Exemplo	Igual a
=	uma = 10	uma = 10

+=	uma += 30	uma = uma + 30
-=	uma -= 15	uma = uma - 15
*=	uma*= 10	uma = uma * 10
/=	uma /= 5	uma = uma / 5
%=	uma% = 5	uma = uma% 5
**=	uma **= 4	uma = uma ** 4
//=	uma //= 5	uma = uma // 5
&=	uma &= 5	uma = uma e 5
|=	uma |= 5	uma = uma | 5
^=	uma ^ = 5	uma = uma ^ 5
>>=	uma >>= 5	uma = uma >> 5
<<=	um <<= 5	uma = uma << 5

Exemplo de operadores de atribuição Python

```
a = 21
b = 10
c = 0
print ("a: {} b: {} c : {}".format(a,b,c))
c = a + b
print ("a: {}  c = a + b: {}".format(a,c))

c += a
print ("a: {} c += a: {}".format(a,c))

c *= a
print ("a: {} c *= a: {}".format(a,c))

c /= a
print ("a: {} c /= a : {}".format(a,c))

c  = 2
```

```
print ("a: {} b: {} c : {}".format(a,b,c))
c %= a
print ("a: {} c %= a: {}".format(a,c))

c **= a
print ("a: {} c **= a: {}".format(a,c))

c //= a
print ("a: {} c //= a: {}".format(a,c))
```

Saída

```
a: 21 b: 10 c : 0
a: 21  c = a + b: 31
a: 21 c += a: 52
a: 21 c *= a: 1092
a: 21 c /= a : 52.0
a: 21 b: 10 c : 2
a: 21 c %= a: 2
a: 21 c **= a: 2097152
a: 21 c //= a: 99864
```

Operadores bit a bit Python

O operador bit a bit trabalha em bits e executa operações bit a bit. Esses operadores são usados para comparar números binários.

Existem os seguintes operadores Bitwise suportados pela linguagem Python

Operador	Nome	Exemplo
&	E	a e b
|	OU	uma | b
^	XOR	uma ^ b
~	NÃO	~a
<<	Preenchimento zero deslocamento à esquerda	um << 3

>>	Mudança à direita assinada	um >> 3

Exemplo de operadores bit a bit Python

```
a = 20
b = 10

print ('a=',a,':',bin(a),'b=',b,':',bin(b))
c = 0

c = a & b;
print ("result of AND is ", c,':',bin(c))

c = a | b;
print ("result of OR is ", c,':',bin(c))

c = a ^ b;
print ("result of EXOR is ", c,':',bin(c))

c = ~a;
print ("result of COMPLEMENT is ", c,':',bin(c))

c = a << 2;
print ("result of LEFT SHIFT is ", c,':',bin(c))

c = a >> 2;
print ("result of RIGHT SHIFT is ", c,':',bin(c))
```

Saída

```
a= 20 : 0b10100 b= 10 : 0b1010
result of AND is  0 : 0b0
result of OR is  30 : 0b11110
result of EXOR is  30 : 0b11110
result of COMPLEMENT is  -21 : -0b10101
result of LEFT SHIFT is  80 : 0b1010000
result of RIGHT SHIFT is  5 : 0b101
```

Operadores Lógicos Python

Operadores lógicos Python são usados para combinar duas ou mais condições e verificar o resultado final. Existem os seguintes operadores lógicos suportados pela linguagem Python. Suponha que a variável a contenha 10 e a variável b contenha 20, então

Operador	Nome	Exemplo
e	E	a e b
ou	OU	a ou B
não	NÃO	não (um)

Exemplo de operadores lógicos Python

```
var = 5

print(var > 3 and var < 10)
print(var > 3 or var < 4)
print(not (var > 3 and var < 10))
```

Saída

```
True
True
False
```

Operadores de associação Python

Os operadores de associação do Python testam a associação em uma sequência, como strings, listas ou tuplas. Existem dois operadores de adesão, conforme explicado abaixo -

Operador	Descrição	Exemplo
em	Retorna True se encontrar uma variável na sequência especificada, caso contrário, retorna false.	uma em b
não em	retorna True se não encontrar uma variável na sequência especificada e	a não em b

	false caso contrário.	

Exemplo de operadores de associação Python

```
a = 10
b = 20
list = [1, 2, 3, 4, 5 ]

print ("a:", a, "b:", b, "list:", list)

if ( a in list ):
   print ("a is present in the given list")
else:
   print ("a is not present in the given list")

if ( b not in list ):
   print ("b is not present in the given list")
else:
   print ("b is present in the given list")

c=b/a
print ("c:", c, "list:", list)
if ( c in list ):
   print ("c is available in the given list")
else:
    print ("c is not available in the given list")
```

Saída

```
a: 10 b: 20 list: [1, 2, 3, 4, 5]
a is not present in the given list
b is not present in the given list
c: 2.0 list: [1, 2, 3, 4, 5]
c is available in the given list
```

Operadores de identidade Python

Os operadores de identidade comparam as localizações de memória de dois objetos. Existem dois operadores de identidade explicados abaixo -

Operador	Descrição	Exemplo
é	Retorna True se ambas as variáveis forem o mesmo objeto e false caso contrário.	a é b
não é	Retorna True se ambas as variáveis não forem o mesmo objeto e false caso contrário.	a não é b

Exemplo de operadores de identidade Python

```
a = [1, 2, 3, 4, 5]
b = [1, 2, 3, 4, 5]
c = a

print(a is c)
print(a is b)

print(a is not c)
print(a is not b)
```

Saída

```
True
False
False
True
```

Precedência de Operadores Python

A tabela a seguir lista todos os operadores da precedência mais alta para a mais baixa.

Sr. Não.	Operador e Descrição
1	** Exponenciação (elevar à potência)

2	**~ + -** Complemento, unário mais e menos (os nomes dos métodos para os dois últimos são +@ e -@)
3	***/% //** Multiplicar, dividir, módulo e divisão de piso
4	**+ -** Adição e subtração
5	**>> <<** Deslocamento bit a bit para a direita e para a esquerda
6	**&** 'E' bit a bit
7	**^ |** `OR' exclusivo bit a bit e `OR' regular
8	**<= < > >=** Operadores de comparação
9	**<> == !=** Operadores de igualdade
10	**=%= /= //= -= += *= **=** Operadores de atribuição
11	**é não é** Operadores de identidade
12	**em não em** Operadores de adesão
13	**não ou e** Operadores lógicos