Python - Operadores de Atribuição

Operador de Atribuição Python

O símbolo = (igual a) é definido como operador de atribuição em Python. O valor da expressão Python à sua direita é atribuído a uma única variável à sua esquerda. O símbolo = como na programação em geral (e no Python em particular) não deve ser confundido com seu uso em matemática, onde afirma que as expressões em ambos os lados do símbolo são iguais.

Exemplo de operador de atribuição em Python

Considere seguir as instruções Python -

```
a = 10
b = 5
a = a + b
print (a)
```

À primeira vista, pelo menos para alguém novo em programação, mas que entende de matemática, a afirmação "a=a+b" parece estranha. Como a poderia ser igual a "a+b"? No entanto, é necessário enfatizar novamente que o símbolo = é um operador de atribuição aqui e não é usado para mostrar a igualdade de LHS e RHS.

Por ser uma atribuição, a expressão à direita é avaliada como 15, o valor é atribuído a a.

Na instrução "a+=b", os dois operadores "+" e "=" podem ser combinados em um operador "+=". É chamado de operador de adição e atribuição. Em uma única instrução, realiza a adição de dois operandos "a" e "b", e o resultado é atribuído ao operando à esquerda, ou seja, "a".

Operadores de Atribuição Aumentada em Python

Além do operador de atribuição simples, Python fornece mais alguns operadores de atribuição para uso avançado. Eles são chamados de operadores de atribuição cumulativos ou aumentados. Neste capítulo, aprenderemos a usar operadores de atribuição aumentada definidos em Python.

Python possui operadores de atribuição aumentados para todos os operadores aritméticos e de comparação .

Os operadores de atribuição aumentada do Python combinam adição e atribuição em uma instrução. Como o Python suporta aritmética mista, os dois operandos podem ser de tipos diferentes. Porém, o tipo do operando esquerdo muda para o operando da direita, se for mais largo.

Exemplo

O operador += é um operador aumentado. Também é chamado de operador de adição cumulativa, pois adiciona "b" em "a" e atribui o resultado de volta a uma variável.

A seguir estão os operadores de atribuição aumentada em Python:

- Operador de adição aumentada
- Operador de Subtração Aumentada
- Operador de multiplicação aumentada
- Operador de Divisão Aumentada
- Operador de Módulo Aumentado
- Operador de Expoente Aumentado
- Operador de Divisão de Piso Aumentado

Operador de adição aumentada (+=)

Os exemplos a seguir ajudarão a entender como funciona o operador "+=" -

```
a=10
b=5
print ("Augmented addition of int and int")
a+=b # equivalent to a=a+b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented addition of int and float")
a+=b  # equivalent to a=a+b
print ("a=",a, "type(a):", type(a))
```

```
a=10.50
b=5+6j
print ("Augmented addition of float and complex")
a+=b #equivalent to a=a+b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented addition of int and int
a= 15 type(a): <class 'int'>
Augmented addition of int and float
a= 15.5 type(a): <class 'float'>
Augmented addition of float and complex
a= (15.5+6j) type(a): <class 'complex'>
```

Operador de subtração aumentada (-=)

Use o símbolo -= para realizar operações de subtração e atribuição em uma única instrução. A instrução "a-=b" executa a atribuição "a=ab". Os operandos podem ser de qualquer tipo numérico. Python executa conversão implícita de tipo no objeto de tamanho mais estreito.

```
a=10
b=5
print ("Augmented subtraction of int and int")
a-=b #equivalent to a=a-b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented subtraction of int and float")
a-=b #equivalent to a=a-b
print ("a=",a, "type(a):", type(a))

a=10.50
b=5+6j
print ("Augmented subtraction of float and complex")
a-=b #equivalent to a=a-b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented subtraction of int and int
a= 5 type(a): <class 'int'>
Augmented subtraction of int and float
a= 4.5 type(a): <class 'float'>
Augmented subtraction of float and complex
a= (5.5-6j) type(a): <class 'complex'>
```

Operador de multiplicação aumentada (*=)

O operador "*=" funciona com princípio semelhante. "a*=b" executa operações de multiplicação e atribuição e é equivalente a "a=a*b". No caso de multiplicação aumentada de dois números complexos, a regra de multiplicação discutida no capítulo anterior é aplicável.

```
a=10
b=5
print ("Augmented multiplication of int and int")
a*=b #equivalent to a=a*b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented multiplication of int and float")
a*=b #equivalent to a=a*b
print ("a=",a, "type(a):", type(a))

a=6+4j
b=3+2j
print ("Augmented multiplication of complex and complex")
a*=b #equivalent to a=a*b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented multiplication of int and int
a= 50 type(a): <class 'int'>
Augmented multiplication of int and float
a= 55.0 type(a): <class 'float'>
Augmented multiplication of complex and complex
a= (10+24j) type(a): <class 'complex'>
```

Operador de Divisão Aumentada (/=)

O símbolo de combinação "/=" atua como operador de divisão e atribuição, portanto "a/=b" é equivalente a "a=a/b". A operação de divisão de operandos int ou float é float. A divisão de dois números complexos retorna um número complexo. Abaixo estão exemplos de operadores de divisão aumentada.

```
a=10
b=5
print ("Augmented division of int and int")
a/=b #equivalent to a=a/b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented division of int and float")
a/=b #equivalent to a=a/b
print ("a=",a, "type(a):", type(a))

a=6+4j
b=3+2j
print ("Augmented division of complex and complex")
a/=b #equivalent to a=a/b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented division of int and int
a= 2.0 type(a): <class 'float'>
Augmented division of int and float
a= 1.8181818181818181 type(a): <class 'float'>
Augmented division of complex and complex
a= (2+0j) type(a): <class 'complex'>
```

Operador de Módulo Aumentado (%=)

Para realizar operações de módulo e atribuição em uma única instrução, use o operador %=. Assim como o operador mod, sua versão aumentada também não é compatível com números complexos.

```
a=10
b=5
print ("Augmented modulus operator with int and int")
a%=b #equivalent to a=a%b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented modulus operator with int and float")
a%=b #equivalent to a=a%b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented modulus operator with int and int
a= 0 type(a): <class 'int'>
Augmented modulus operator with int and float
a= 4.5 type(a): <class 'float'>
```

Operador Expoente Aumentado (**=)

O operador "**=" resulta no cálculo de "a" elevado a "b" e na atribuição do valor de volta a "a". Abaixo estão alguns exemplos -

```
a=10
b=5
print ("Augmented exponent operator with int and int")
a**=b #equivalent to a=a**b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented exponent operator with int and float")
a**=b #equivalent to a=a**b
print ("a=",a, "type(a):", type(a))

a=6+4j
b=3+2j
print ("Augmented exponent operator with complex and complex")
a**=b #equivalent to a=a**b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented exponent operator with int and int
a= 100000 type(a): <class 'int'>
Augmented exponent operator with int and float
a= 316227.7660168379 type(a): <class 'float'>
Augmented exponent operator with complex and complex
a= (97.52306038414744-62.22529992036203j) type(a): <class 'complex'>
```

Operador de divisão de piso aumentado (//=)

Para realizar divisão e atribuição de piso em uma única instrução, use o operador "//=". "a//=b" é equivalente a "a=a//b". Este operador não pode ser usado com números complexos.

```
a=10
b=5
print ("Augmented floor division operator with int and int")
a//=b #equivalent to a=a//b
print ("a=",a, "type(a):", type(a))

a=10
b=5.5
print ("Augmented floor division operator with int and float")
a//=b #equivalent to a=a//b
print ("a=",a, "type(a):", type(a))
```

Ele produzirá a seguinte **saída** -

```
Augmented floor division operator with int and int
a= 2 type(a): <class 'int'>
Augmented floor division operator with int and float
a= 1.0 type(a): <class 'float'>
```