100 XP

Variáveis e tipos de dados básicos em Python

4 minutos

As variáveis são uma das bases fundamentais de programas escritos em Python. Variáveis contêm dados na memória. Elas têm nomes e podem ser referenciadas por esses nomes. As variáveis também têm *tipos*, que especificam o tipo de dados que podem armazenar (como cadeias de caracteres e inteiros) e podem ser usadas em expressões que usam *operadores* (como `+` e `-`) para manipular os valores delas.

Variáveis

No Python, uma variável é declarada e recebe um valor usando o operador de atribuição `=`. A variável está no lado esquerdo do operador, e o valor que está sendo atribuído – que pode ser uma expressão como `2 + 2` e até mesmo incluir outras variáveis – está no lado direito. Por exemplo:

Python

```
x = 1         # assign variable x the value 1
y = x + 5     # assign variable y the value of x plus 5
z = y         # assign variable z the value of y
```

Esses exemplos atribuem números de variáveis, mas números são apenas um dos vários tipos de dados compatíveis com Python. Observe que há nenhum tipo declarado para as variáveis. Isso ocorre porque o Python é uma linguagem *dinamicamente tipada*, o que significa que o tipo de variável é determinado pelos dados atribuídos a ela. Nos exemplos acima, as variáveis x, y e z são tipos de inteiro, capazes de armazenar números inteiros positivos e negativos.

Nomes de variáveis diferenciam maiúsculas de minúsculas e podem usar qualquer letra, número e o caractere de sublinhado (_). No entanto, eles não podem começar com um número.

Trabalhando com números

A maioria dos programas manipula números. Computadores tratam números inteiros e números decimais de forma diferente. Considere o seguinte código:

Python

```
x = 1       # integer
x = 1.0     # decimal (floating point)
```

O Python cria números inteiros de um tipo de dados interno chamado `int` e decimais (números de ponto flutuante) como instâncias de `float`. A função `type()` interna do Python retorna um tipo de dados. O seguinte código gera tipos de dados:

Python

```
x = 1
print(type(x)) # outputs: <class 'int'>

x = 1.0
print(type(x)) # outputs: <class 'float'>
```

A adição do ".0" ao final de "1" faz uma grande diferença na forma como a linguagem de programação trata um valor. O tipo de dados afeta como o valor é armazenado na memória, como o processador (CPU) lida com os dados ao avaliar expressões, como os dados se relacionam com outros dados e que tipos de operações podem ser executados com ele.

Trabalhando com boolianos

Outro tipo de dados comum é o tipo booliano, que contém o valor `True` ou `False`:

Python

```
x = True
print(type(x)) # outputs: <class 'bool'>
```

Internamente, `bool` é tratado como um tipo especial de inteiro. Tecnicamente, `True` tem um valor de 1 e `False` tem um valor de 0. Normalmente, boolianos não são usados para executar operações matemáticas; em vez disso, eles são usados para tomar decisões e executar ramificação. No entanto, é interessante entender a relação entre tipos. Muitos tipos são nada mais do que versões especializadas de tipos mais gerais. Inteiros são um subconjunto dos números de ponto flutuante. Boolianos são um subconjunto de inteiros.

Trabalhando com cadeias de caracteres

Junto com números, as cadeias de caracteres estão entre os tipos de dados mais usados. Uma cadeia de caracteres é uma coleção de zero ou mais caracteres. Cadeias de caracteres

normalmente são declaradas usando aspas simples, mas aspas duplas podem ser usadas:

Python

```
x = 'This is a string'
print(x) # outputs: This is a string
print(type(x)) # outputs: <class 'str'>
y = "This is also a string"
```

Você pode adicionar cadeias de caracteres a outras cadeias de caracteres – uma operação conhecida como "concatenação" – com o mesmo operador `+` que adiciona dois números:

Python

```
x = 'Hello' + ' ' + 'World!'
print(x) # outputs: Hello World!
```

Você aprenderá mais sobre cadeias de caracteres em outra lição, incluindo como analisá-las e como manipulá-las de várias maneiras. Você também aprenderá sobre outros tipos de dados importantes, como listas, que armazenam coleções de dados e frequentemente são usadas para conter coleções de cadeias de caracteres.

Imprimir para o console

No Python, a função `print`, que é uma das mais de 60 funções integradas à linguagem, gera texto na tela.

A instrução a seguir exibe "Olá, Mundo!" na tela:

Python

```
print('Hello World!')
```

O argumento passado para `print` é uma *cadeia de caracteres*, que é um dos tipos de dados fundamentais em Python usados para armazenar e gerenciar texto. Por padrão, `print` produz um caractere de nova linha no final da linha para que chamadas subsequentes para `print` comecem na próxima linha.

Todas as unidades foram concluídas:

Concluir módulo