100 XP

Trabalhar com números no Python

4 minutos

Além da aritmética básica, você pode usar outras operações em números. Pode ser necessário realizar o arredondamento ou converter cadeia de caracteres em números.

No cenário deste módulo, você quer aceitar a entrada de um usuário. A entrada é uma cadeia de caracteres, portanto, você precisará convertê-la em um número. Além disso, o usuário pode inserir valores que forneçam uma resposta negativa, que você não deseja exibir. Talvez seja necessário converter a resposta para o valor absoluto. Felizmente, o Python fornece utilitários para essas operações.

Converter cadeias de caracteres em números

O Python dá suporte a dois tipos principais de números: inteiros (ou `int`) e de ponto flutuante (ou `float`). A principal diferença entre os dois é a existência de um ponto decimal; inteiros são números inteiros, enquanto pontos flutuantes contêm um valor decimal.

Ao converter cadeia de caracteres em números, você indica o tipo de número que deseja criar. Convém decidir se você precisa de um ponto decimal. Use `int` para converter para inteiro, e `float`, para converter para um número de ponto flutuante.

Python

```
demo_int = int('215')
print(demo_int)

demo_float = float('215.3')
print(demo_float)
```

Output

```
215
215.3
```

> **Importante**
>
> Se você inserir um valor inválido em `int` ou `float`, receberá um erro.

Valores absolutos

Em matemática, um valor absoluto é um número não negativo sem seu sinal. Usar um valor absoluto pode ser útil em diferentes situações, incluindo nosso exemplo para determinar a distância entre dois planetas. Considere o seguinte cálculo:

Python

```
print(39 - 16)
print(16 - 39)
```

Observe que a diferença entre as duas equações é que os números são opostos. As respostas são `23` e `-23`, respectivamente. Quando você está determinando a distância entre dois planetas, a ordem em que você digita os números não importa, pois a resposta absoluta será a mesma.

Converta o valor negativo para ser seu valor absoluto usando `abs`. Se você executar a mesma operação usando `abs` (e imprimir as respostas), observará que mostrará `23` nas duas equações.

Python

```
print(abs(39 - 16))
```

```
print(abs(16 - 39))
```

Output

```
23
23
```

Arredondamento

A função interna do Python chamada `round` também é útil. Use-o para arredondar para o número inteiro mais próximo se o valor decimal for maior que `.5`, ou para baixo se for menor que `.5`. Se o valor decimal for igual a `.5`, a função arredonda para cima ou para baixo para o inteiro **par** mais próximo.

Python

```
print(round(1.4))
print(round(1.5))
print(round(2.5))
print(round(2.6))
```

Output

```
1
2
2
3
```

Biblioteca de matemática

O Python tem bibliotecas para fornecer operações e cálculos mais avançados. Uma das mais comuns é a biblioteca `math`. A `math` permite que você execute o arredondamento com valores de `floor` e `ceil`, forneça o valor de pi e várias outras operações. Vamos ver como usar essa biblioteca para arredondar para cima ou para baixo.

O arredondamento de números permite remover a parte decimal de um ponto flutuante. Você pode optar por sempre arredondar o número inteiro mais próximo para cima com `ceil`, ou para baixo com `floor`.

Python

```
from math import ceil, floor

round_up = ceil(12.5)
print(round_up)

round_down = floor(12.5)
print(round_down)
```

Output

```
13
12
```

Unidade seguinte: Exercício – converter cadeias de caracteres em números e usar valores absolutos

Continuar >