Float

Floats são parecidos com inteiros mas além de armazenar a parte inteira do número eles também podem armazenar o ponto flutuante, a fração, e são usados para armazenar resultados que não podem ser armazenados em inteiros, por exemplo.

>>> valor = 5 / 2 # cindo dividido por 2

>>> print(valor)

2.5

A presença de um . em um número faz com que o Python entenda que queremos armazena-lo em um objeto da classe float e assim como os inteiros ela possui todos os métodos especiais dunder para os protocolos que implementa e também métodos que são particulares apenas dos números floats.

Exemplos de uso de um float

# Resultados de divisão

valor = 5 / 2

# Coordenadas geográficas

latitude = -37.80467681

longitude = 144.9659498

# Saldo de pontuação (em jogos por exempo)

pontos = 355.8

**NOTA** Para trabalhar com dados monetários (dinheiro) damos preferência a um tipo especializado chamado Decimal ao invés de float mas não se preocupe que abordaremos isso em breve.

Booleanos

O tipo booleano é representado pela classe bool e ele pode armazenar apenas 2 estados Verdadeiro e Falso, em teoria poderiamos aplicar aqui a lógica binária e em nosso programa dizer que 0 é falso enquanto 1 é verdadeiro, e de fato é isso que Python faz por debaixo dos panos, porém para ficar com uma sintaxe mais bonita termos o tipo bool e suas variações True e False.

Quando utilizamos esse tipo? sempre que precisamos de **flags**, variavéis que podem estar em um desses dois estados, veja alguns exemplos:

# Tornar um usuário administrador

is\_admin = True

# Verificar se o usuário quer continuar uma operação

continuar = False

# Definir se um produto está ativo em uma loja

active = True

Apesar de ser bastante simples, o tipo bool é muito útil e ele por sí só forma um protocolo chamado Boolean, com objetos booleanos podemos criar expressões condicionais, como as que criamos em nosso script hello.py

if current\_language == "pt\_BR":

msg = "Hello, World!"

A parte current\_language == "pt\_BR" retorna um valor do tipo bool e sempre que usamos o statement if a expressão em seguida precisa obrigatoriamente retornar um objeto que tenha o protocolo Boolean assim como esse.

Veja em seu terminal:

# inicialize a variável

>>> current\_language = "en\_US"

# obtenha um bool através de comparação por igualdade

>>> current\_language == "pt\_BR"

False

# verifique o tipo diretamente

>>> type(current\_language == "pt\_BR")

bool

# Isso também funciona com números int

>>> type(1 == 1)

True

Se você rodar o comando dir(int) verá que na lista de métodos especiais tem um chamado \_\_bool\_\_ e é ele que é chamado quando fazemos operações if usando os inteiros.

if 500:

print("Ok, 500 é um int que implementa \_\_bool\_\_")

Muitos objetos no Python implementam \_\_bool\_\_ e podem ser usados diretamente após o if mesmo que não exista uma expressão de comparação.

NoneType

Em alguns casos precisamos inicializar uma variável porém ainda não temos o valor para armazenar nela, nesse caso usamos o objeto None

>>> type(None)

NoneType

Este é um tipo especial que serve para quando não possuimos um valor mas precisamos da variável definida pois em algum momento no decorrer do programa iremos refazer a atribuição daquela variável.

produto = None

if produto is None:

produto = funcao\_para\_definicao\_do\_produto()

O objeto None é um singleton, só existe um None mesmo que você defina várias variáveis como None todas elas farão referência ao mesmo None

>>> a = None

>>> b = None

>>> id(a)

139862040616512

>>> id(b)

139862040616512

>>> a is b

True

a == b

True

**Exercicio**

Vamos criar um programa que imprime as tabuadas de 1 até 10