

# ASIMOV

---

## Tuplas

Em Python, as tuplas são muito semelhantes às listas, no entanto, ao contrário das listas, elas são *imutáveis*, o que significa que elas não podem ser alteradas. Você usaria tuplas para apresentar coisas que não deveriam ser alteradas, como dias da semana ou datas em um calendário.

Nesta seção, obteremos uma breve visão geral do seguinte:

1.) Construindo Tuplas    2.) Métodos básicos das Tuplas    3.) Imutabilidade    4.) Quando usar Tuplas

Você terá uma intuição de como usar tuplas com base no que você aprendeu sobre as listas. Nós podemos tratá-los de forma muito semelhante, com a maior distinção é que as tuplas são imutáveis.

## Construindo Tuplas

A construção de tuplas usa () com elementos separados por vírgulas. Por exemplo:

```
In [1]: # Pode-se criar uma tupla com múltiplos elementos
t = (1,2,3)
```

```
In [6]: # O método len() funciona também para tuplas
len(t)
```

Out[6]: 3

```
In [8]: # Você também pode variar os tipos de dados
t = ('one',2)

# Mostra
t
```

Out[8]: ('one', 2)

```
In [4]: # E a indexação funciona exatamente como nas listas
t[0]
```

Out[4]: 'one'

```
In [11]: # Corte de dados também...
t[-1]
```

```
Out[11]: 2
```

## Métodos básicos da Tupla

As tuplas têm métodos internos, mas não tantas quanto listas. Vamos ver dois deles:

```
In [12]: # Use .index com o valor de parâmetro para retornar o índice do mesmo  
t.index('one')
```

```
Out[12]: 0
```

```
In [13]: # Use .count() para saber quantas vezes determinado elemento apareceu na tupla  
t.count('one')
```

```
Out[13]: 1
```

## Imutabilidade

Como mencionado anteriormente, tuplas são imutáveis:

```
In [14]: t[0] = 'change'
```

```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-14-93def5f9b4bd> in <module>()  
----> 1 t[0] = 'change'
```

```
TypeError: 'tuple' object does not support item assignment
```

Por causa dessa imutabilidade, as tuplas não podem crescer. Uma vez que uma tupla é feita, não podemos adicionar a ela.

```
In [15]: t.append('nope')
```

```
-----  
AttributeError                            Traceback (most recent call last)  
<ipython-input-15-799b3447c4d9> in <module>()  
----> 1 t.append('nope')
```

```
AttributeError: 'tuple' object has no attribute 'append'
```

## Quando usar tuplas

Você pode estar se perguntando: "Por que se preocupar em usar tuplas quando eles têm menos métodos disponíveis?" Para ser honesto, as tuplas não são usadas tantas vezes como listas na programação, mas são usadas quando a imutabilidade é necessária. Se no seu programa você está passando por um objeto e precisa ter certeza de que ele não seja alterado, então a tupla se tornará sua solução. Ele fornece uma fonte conveniente de integridade de dados.

Agora você pode criar e usar suas tuplas em sua programação, bem como ter uma compreensão da sua imutabilidade.

