TUPLAS EM PYTHON (, ,)

Guia completo com exemplos, resultados e explicacoes

Prof. Gustavo Franz (Science/Biology)
Python Developer in Progress
Building Educational Solutions with Python

GitHub: github.com/GustaFranz

Regras principais

- ✓ Tuplas sao **IMUTAVEIS**: nao da para alterar, adicionar ou remover elementos depois de criadas.
- ✓ Ordenadas: os elementos possuem posicao (indice).
- ✓ Permitem elementos duplicados.
- ✓ Podem armazenar diferentes tipos de dados.
- ✓ Usamos parenteses ( ) para criar tuplas.

Criando tuplas

Formas de criar uma tupla

```
t1 = (10, 20, 30)  # tupla com numeros
t2 = ('Python', 3.14, True)  # tipos diferentes
t3 = (1,)  # 1 elemento (virgula obrigatoria)
t4 = tuple([1, 2, 3, 4])  # convertendo lista em tupla
```

Observacao importante

(1) NAO e uma tupla — e apenas um valor entre parenteses. A tupla de um elemento exige a virgula: (1,).

Operacoes e analise de tuplas

Exemplos considerando **numeros = (10, 20, 30, 40, 50)**.

Comando	Resultado	Por que esse e o resultado?
`numeros[0]`	`10`	O indice 0 e o primeiro elemento da tupla.
`numeros[2]`	`30`	O indice 2 aponta para o terceiro elemento.
`numeros[-1]`	`50`	Indices negativos comecam pelo final: -1 e o ultimo.
`numeros[-2]`	`40`	O -2 e o penultimo elemento.
`numeros[1:4]`	`(20, 30, 40)`	Do indice 1 (incluido) ate antes do 4.
`numeros[0:3]`	`(10, 20, 30)`	Os indices 0, 1 e 2, parando antes do 3.
`numeros[2:]`	`(30, 40, 50)`	Do indice 2 ate o ultimo elemento.

Comando	Resultado	Por que esse e o resultado?
`numeros[:3]`	`(10, 20, 30)`	Do primeiro ate antes do indice 3.
`numeros[:]`	`(10,...,50)`	Todos os elementos (copia da tupla).
`numeros[-3:]`	`(30, 40, 50)`	O indice -3 e o valor 30; vai ate o final.
`len(numeros)`	`5`	len() conta quantos elementos existem na tupla.
`30 in numeros`	`True`	in verifica se o valor 30 esta na tupla.
`100 in numeros`	`False`	O valor 100 nao existe na tupla.
`numeros.count(20)`	`1`	count() informa quantas vezes o valor aparece.
`numeros.index(40)`	`3`	index() retorna o indice da primeira ocorrencia.
`numeros + (60, 70)`	`(...,60,70)`	Cria uma nova tupla unindo as duas; a original nao muda.
`numeros * 2`	`(...repetida)`	Repete a sequencia inteira da tupla duas vezes.

Fatiamento (slice) visual

Considere **numeros = (10, 20, 30, 40, 50)** e seus indices:

Indice	0	1	2	3	4
`Valor`	10	20	30	40	50

Expressao	Resultado	Leitura
`numeros[1:4]`	`(20, 30, 40)`	Do indice 1 (incluido) ate antes do 4.
`numeros[2:]`	`(30, 40, 50)`	Do indice 2 ate o final.
`numeros[:3]`	`(10, 20, 30)`	Do inicio ate antes do indice 3.
`numeros[-3:]`	`(30, 40, 50)`	Os tres ultimos elementos.

Regra de ouro do fatiamento

O indice inicial e INCLUIDO; o indice final NAO e incluido.

Exercicio: medias e situacao dos alunos

Voce recebeu os dados de alunos e suas notas. Calcule a media de cada um e exiba nome, notas, media e situacao (Aprovado se media >= 7; senao Reprovado).

medias.py

```
alunos = (
    ('Ana',    (8.5, 7.0, 9.0)),
    ('Bruno',  (6.0, 5.5, 7.0)),
    ('Carla',  (9.5, 9.0, 8.5)),
    ('Daniel', (7.0, 7.5, 6.5)),
)

resultado = []
for nome, notas in alunos:
    media = sum(notas) / len(notas)
    situacao = 'Aprovado' if media >= 7 else 'Reprovado'
    resultado.append((nome, round(media, 2), situacao))

for nome, media, situacao in resultado:
    print(nome, media, situacao)
```

Aluno	Notas	Media	Situacao
Ana	`(8.5, 7.0, 9.0)`	`8.17`	Aprovado
Bruno	`(6.0, 5.5, 7.0)`	`6.17`	Reprovado
Carla	`(9.5, 9.0, 8.5)`	`9.00`	Aprovado
Daniel	`(7.0, 7.5, 6.5)`	`7.00`	Aprovado

Dica final

Tuplas sao ideais para dados que NAO devem mudar — como coordenadas, configuracoes e registros fixos.

Prof. Gustavo Franz (Science/Biology)
Python Developer in Progress
Building Educational Solutions with Python

Professor de Ciencias e Biologia desde 2013 — atualmente estudando Programacao.

Eu crio estes resumos para organizar os assuntos e estudar de forma clara e objetiva. Estou compartilhando porque eles tambem podem ajudar outros desenvolvedores que estao comecando. Bons estudos — vamos aprender juntos!

GitHub: github.com/GustaFranz

Exercicios Python: github.com/GustaFranz/exercicios_python
Para quem quiser ver a resolucao dos meus exercicios em Python.

EasyAnsi: github.com/GustaFranz/easyansi
Para quem quiser codificar personalizando com cores e estilos de forma mais facil, pratica e intuitiva.