

**Educação  
Profissional  
Paulista**

Técnico em  
**Ciência de  
Dados**

# Variáveis e tipos de dados

## Dicionários

### Aula 1

[DADOS]ANO1C2B2S12A1

# Exposição



## Objetivos da Aula

- Introduzir o conceito de dicionários em Python.



## Competências da Unidade (Técnicas e Socioemocionais)

- Ser proficiente em linguagens de programação para manipular e analisar grandes conjuntos de dados;
- Usar técnicas para explorar e analisar dados, aplicar modelos estatísticos, identificar padrões, realizar inferências e tomar decisões baseadas em evidências;
- Colaborar, efetivamente, com outros profissionais, como cientistas de dados e engenheiros de dados;
- Trabalhar em equipes multifuncionais colaborando com colegas, gestores e clientes.



## Recursos Didáticos

- Recurso audiovisual para exibição de vídeos e imagens;
- Acesso ao laboratório de informática e/ou internet;
- Software Anaconda/Jupyter Notebook instalado, ou similar.



## Duração da Aula

50 minutos

## Exposição

# Motivação: jogo de RPG – Inventário de jogador



© Getty Images

Imagine que você está desenvolvendo um jogo de **RPG** (*Role-playing game*) **online**, em que os jogadores têm inventários para armazenar itens como armas, poções, armaduras etc. Cada jogador pode ter um **dicionário de inventário**, no qual as chaves representam os tipos de itens, e os valores representam a **quantidade** de cada item.

Que estrutura de dados em Python você usaria para armazenar os dados?



Exposição

## Dicionário



© Getty Images

Um **dicionário em Python** é uma **estrutura de dados** que permite **armazenar pares chave-valor**. Cada chave deve ser única e associada a um valor específico.

Os dicionários são conhecidos por sua capacidade de associar chaves a valores, permitindo que você **recupere** facilmente o valor associado a uma determinada chave.

Essa recuperação facilitada é o principal diferenciador dos dicionários em relação a outras estruturas de dados, aumentando a performance do código na consulta de informações.

## Dicionário

A sintaxe básica de um dicionário inclui chaves - `{ }` - para delimitar os elementos e usar pares chave-valor separados por dois pontos:

```
meu_dicionario = {'chave1': 'valor1', 'chave2': 'valor2', ...}
```

```
print(type(meu_dicionario))  
<class 'dict'>
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Exemplo

De acordo com a definição, qual dos exemplos abaixo é um dicionário?

```
# Exemplos

exemplo1 = [1, 2, 3, 4]
exemplo2 = {1, 2, 3}
exemplo3 = {'key': 'value', 'name': 'John'}
exemplo4 = (10, 20, 30)
exemplo5 = 5.5
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Criando um dicionário

### 1. Sintaxe de chaves e valores:

```
: # Criando um dicionário vazio
meu_dicionario = {}

# Adicionando pares chave-valor
meu_dicionario['nome'] = 'João'
meu_dicionario['idade'] = 25
meu_dicionario['cidade'] = 'São Paulo'

print(meu_dicionario)

{'nome': 'João', 'idade': 25, 'cidade': 'São Paulo'}
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.



## Criando um dicionário

### 2. Usando a função dict():

```
# Criando um dicionário usando a função dict()
outro_dicionario = dict(nome='Maria', idade=30, cidade='Campinas')

print(outro_dicionario)
```

```
{'nome': 'Maria', 'idade': 30, 'cidade': 'Campinas'}
```

### 3. Atribuindo dicionários diretamente:

```
# Criando um dicionário diretamente
dicionario_direto = {'a': 1, 'b': 2, 'c': 3}

print(dicionario_direto)
```

```
{'a': 1, 'b': 2, 'c': 3}
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Acessando dicionários

### 1. Acesso direto

Os elementos do dicionário podem ser acessados através de suas chaves. O acesso é feito utilizando a notação de colchetes: **[]**.

```
meu_dicionario = {'nome': 'João', 'idade': 25, 'cidade': 'São Paulo'}  
  
# Acessando o valor associado à chave 'idade'  
idade = meu_dicionario['idade']  
print(idade)
```

25

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Acessando dicionários

### 2. Método `get()`

O método **`get()`** permite acessar um valor com uma chave e fornece um valor padrão se a chave não existir.

```
cidade = meu_dicionario.get('cidade', 'Cidade não especificada')
print(cidade)

# Caso a chave não exista
profissao = meu_dicionario.get('profissao', 'Não especificada')
print(profissao)
```

```
São Paulo
Não especificada
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Acessando dicionários

### 3. Iterando pelas chaves

Você pode iterar pelas chaves e acessar os valores correspondentes:

```
for chave in meu_dicionario:  
    valor = meu_dicionario[chave]  
    print(f'{chave}: {valor}')
```

```
nome: João  
idade: 25  
cidade: São Paulo
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

## Acessando dicionários

### 4. Método `keys()` e iteração

O método **`keys()`** retorna uma visão das chaves do dicionário, que pode ser usada para iterar:

```
for chave in meu_dicionario.keys():  
    valor = meu_dicionario[chave]  
    print(f'{chave}: {valor}')
```

```
nome: João  
idade: 25  
cidade: São Paulo
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.



## Acessando dicionários

### 5. Método `items()`

O método **`items()`** retorna pares chave-valor como tuplas e pode ser usado em um loop:

```
for chave, valor in meu_dicionario.items():  
    print(f'{chave}: {valor}')
```

```
nome: João  
idade: 25  
cidade: São Paulo
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.

# Como criar e acessar o inventário do jogo de RPG?

Imagine que você está desenvolvendo um jogo de **RPG** (*Role-playing game*) **online**, em que os jogadores têm inventários para armazenar itens como armas, poções, armaduras etc. Cada jogador pode ter um **dicionário de inventário**, no qual as chaves representam os tipos de itens e os valores representam a **quantidade** de cada item. Que estrutura de dados em Python você usaria para armazenar os dados?

```
# Dicionário de Inventário de Jogador
inventario_jogador = {'armas': 3, 'poções': 5, 'armaduras': 2, 'moedas': 100}

# Acesso aos itens do inventário
armas_do_jogador = inventario_jogador['armas']
poções_do_jogador = inventario_jogador['poções']
```

Elaborado especialmente para o curso com a ferramenta Jupyter Notebook.



Vamos  
fazer um  
**quiz**

**Qual é a maneira correta de acessar o valor associado à chave 'idade' em um dicionário chamado dados?**

`dados(idade)`

`dados['idade']`

`dados.valor('idade')`

`dados.take('idade')`





Vamos  
fazer um  
**quiz**

Qual é a maneira correta de acessar o valor associado à chave 'idade' em um dicionário chamado dados?



`dados(idade)`

`dados['idade']`



`dados.valor('idade')`

`dados.take('idade')`



### FEEDBACK GERAL DA ATIVIDADE

Em Python, você acessa o valor associado a uma chave em um dicionário usando colchetes []. A sintaxe correta para acessar o valor associado à chave 'idade' em um dicionário chamado dados é **`dados['idade']`**.



Vamos  
fazer um  
**quiz**

**Qual das seguintes opções é uma maneira válida de criar um dicionário em Python?**

```
novo_dicionario = {1, 2, 3, 4}
```

```
novo_dicionario = dict(1='um',  
2='dois', 3='três')
```

```
novo_dicionario = {'nome': 'Alice',  
'idade': 25, 'cidade': Santos}
```

```
novo_dicionario = dict(['nome', 'Alice',  
'idade', 25, 'cidade', Santos])
```





Vamos  
fazer um  
**quiz**

**Qual das seguintes opções é uma maneira válida de criar um dicionário em Python?**



```
novo_dicionario = {1, 2, 3, 4}
```

```
novo_dicionario = dict(1='um',  
2='dois', 3='três')
```



```
novo_dicionario = {'nome': 'Alice',  
'idade': 25, 'cidade': Santos}
```

```
novo_dicionario = dict(['nome', 'Alice',  
'idade', 25, 'cidade', Santos])
```



### FEEDBACK GERAL DA ATIVIDADE

A alternativa correta é `novo_dicionario = {'nome': 'Alice', 'idade': 25, 'cidade': Santos}`. Essa é a sintaxe correta para criar um dicionário em Python usando chaves e pares chave-valor.



Vamos  
fazer um  
**quiz**

## O que é um dicionário em Python?

Uma sequência ordenada de elementos.

Uma coleção imutável de pares chave-valor.

Uma estrutura de dados que armazena apenas números inteiros.

Uma coleção mutável e indexada de pares chave-valor.





Vamos  
fazer um  
**quiz**

## O que é um dicionário em Python?



Uma sequência ordenada de elementos.

Uma coleção imutável de pares chave-valor.



Uma estrutura de dados que armazena apenas números inteiros.

Uma coleção mutável e indexada de pares chave-valor.



### FEEDBACK GERAL DA ATIVIDADE

Em Python, um dicionário é uma estrutura de dados que permite armazenar pares únicos de chave-valor, mutável (pode ser modificado após a criação) e indexado (os valores podem ser acessados por meio das chaves).



© Getty Images

O que nós  
**aprendemos  
hoje?**

## Hoje desenvolvemos:

- 1** Conhecimento sobre o conceito de dicionário em Python.
- 2** Experiência prática de criação e acesso a dicionários em Python.



# Saiba mais

Quer saber mais sobre dicionário em Python? Clique nos links abaixo e se aprofunde neste conceito.

SILVEIRA, G. *Python collections parte 2: conjuntos e dicionários*. Alura, 2023. Disponível em:

<https://cursos.alura.com.br/course/python-collections-conjuntos-e-dicionarios/task/53514>. Acesso em: 13 mar. 2024.

ORESTES, Y. *Python: Trabalhando com dicionários*. Alura, 2018. Disponível em:

<https://www.alura.com.br/artigos/trabalhando-com-o-dicionario-no-python>. Acesso em: 13 mar. 2024.



# Referências da aula

MENEZES, N. N. C. *Introdução à programação com Python: algoritmos e lógica de programação para iniciantes*. São Paulo: Novatec, 2019.

ORESTES, Y. Python: Trabalhando com dicionários. Alura, 2018. Disponível em: <https://www.alura.com.br/artigos/trabalhando-com-o-dicionario-no-python>. Acesso em: 13 mar. 2024.

SILVEIRA, G. *Python collections parte 2: conjuntos e dicionários*. Alura, 2023. Disponível em: <https://cursos.alura.com.br/course/python-collections-conjuntos-e-dicionarios/task/53514>. Acesso em: 13 mar. 2024.

Identidade Visual: imagens © Getty Images

# Educação Profissional Paulista

Técnico em  
**Ciência de  
Dados**