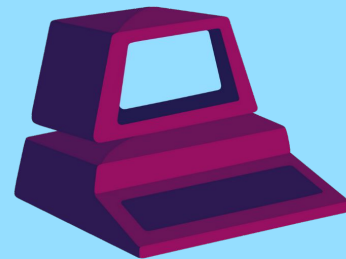


Aula 6 - Turma Avançado

- Dicionários

PET



COMPUTAÇÃO

2026

O que é um Dicionário?

- Dicionários são coleções de dados não ordenadas e mutáveis.
 - Armazenam dados como pares de **chave e valor**.
 - Cada chave deve ser única e imutável (como strings, números ou tuplas).
 - Os valores podem ser de qualquer tipo de dado e podem se repetir.
-
- Pense em uma agenda de contatos: o nome é a chave e o número de telefone é o valor.

Para o que servem?

- Organização e Legibilidade: Permitem que você acesse dados por um nome descritivo (a chave), não por um índice numérico.
- Acesso Rápido: A busca por uma chave é extremamente rápida, ideal para grandes conjuntos de dados.
- São perfeitos para representar objetos do mundo real, como um usuário, um produto em um e-commerce ou um registro de banco de dados.

Exemplo de Dicionário



```
1 aluno = {  
2     "nome": "Ana",  
3     "idade": 22,  
4     "curso": "Ciência da Computação"  
5 }  
6  
7 # Acessando um valor pela chave  
8 print(aluno["nome"])
```

2.

Métodos de dicionários

Acessando Elementos

- Para acessar um valor, use a chave entre colchetes []. Se a chave não existir, ocorre um erro.
- Uma alternativa segura é o método get(), que retorna o valor de uma chave especificada. Se a chave não existir, ele retorna None ou um valor padrão que você definir.



```
1  aluno = {"nome": "Ana", "idade": 22}
2
3  # Acesso direto (pode gerar erro)
4  print(aluno["nome"])
5
6  # Acesso seguro com .get()
7  print(aluno.get("cidade", "Informação não disponível"))
```

Adicionando ou Modificando Itens

- O método `update()` adiciona um novo par chave-valor ou atualiza o valor de uma chave existente.

```
1 aluno = {"nome": "Ana", "idade": 22}
2
3 # Adiciona uma nova chave
4 aluno.update({"cidade": "São Paulo"})
5
6 # Atualiza o valor de uma chave existente
7 aluno.update({"idade": 23})
8
9 print(aluno)
```

Saída:

{'nome': 'Ana', 'idade': 23, 'cidade': 'São Paulo'}

Contando Elementos

- A função `len()` retorna o número total de itens (pares chave-valor) em um dicionário.



```
1 aluno = {"nome": "Ana", "idade": 23, "cidade": "São Paulo"}
2
3 print(len(aluno))
```

- Saída: 3

Checando se uma Chave Existe

- O operador `in` verifica se uma chave está presente no dicionário. Ele retorna `True` ou `False`.



```
1 aluno = {"nome": "Ana", "idade": 23}
2
3 print("nome" in aluno)
4 print("curso" in aluno)
```

Saída:

True

False

Listando Chaves e Valores

- O método `.keys()` retorna uma "visão" de todas as chaves.
- O método `.values()` retorna uma "visão" de todos os valores.



```
1 aluno = {"nome": "Ana", "idade": 23}
2
3 print(aluno.keys())
4 print(aluno.values())
```

Saída:

```
dict_keys(['nome', 'idade'])
dict_values(['Ana', 23])
```

Ordenando Elementos

- A função `sorted()` retorna uma nova lista contendo as chaves do dicionário em ordem alfabética (ou numérica). Ela não altera o dicionário original.

```
1 aluno = {"nome": "Ana", "idade": 23, "cidade": "São Paulo"}  
2  
3 chaves_ordenadas = sorted(aluno.keys())  
4  
5 print(chaves_ordenadas)
```

- Saída: `['cidade', 'idade', 'nome']`

Removendo Elementos

- O método `.pop()` remove um item pela sua chave e retorna o valor associado àquela chave.

```
1 aluno = {"nome": "Ana", "idade": 23, "cidade": "São Paulo"}
2
3 # Remove o par chave-valor 'idade'
4 idade_removida = aluno.pop("idade")
5
6 print(idade_removida)
7 print(aluno)
```

- Saída: 23 {'nome': 'Ana', 'cidade': 'São Paulo'}

Removendo Elementos

- Para ver saber mais sobre os métodos e funções dos Dicionários no Python, acesse

<https://docs.python.org/pt-br/3.13/tutorial/datastructures.html#dictionaries>

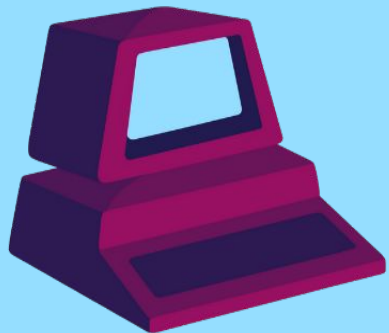
3.

Projeto

Projeto

- Crie um dicionário para armazenar as informações de cada item no estoque: nome, preço e quantidade.
- Mude a lógica de programação para armazenar os dicionários na lista de produtos, e não apenas os nomes.
- **Atenção:** Não se confunda. Cada item deve ser um dicionário e a lista deve conter esses dicionários!

Obrigado!



PET
COMPUTAÇÃO

pet.inf.ufpr.br
pet@inf.ufpr.br
@petcompufpr