Universidade POSITIVO

AULA 5 – DICIONÁRIOS, LISTAS E TUPLAS EM PYTHON (UNIDADE II)

Objetivos da Aula

- Compreender a diferença entre listas, tuplas e dicionários.
- Aprender a criar, acessar e manipular essas estruturas de dados.
- Realizar iterações com estruturas de repetição (for) sobre listas, tuplas e dicionários.
- Aplicar essas estruturas em exemplos práticos com base em problemas do dia a dia.

1. LISTAS

1.1 O que são listas?

- Estrutura **mutável**: você pode alterar, adicionar ou remover elementos após sua criação.
- Armazenam **múltiplos elementos** em uma sequência ordenada.
- Criadas com colchetes `[]`.

Exemplo:

```
nomes = ["Ana", "Bruno", "Carlos"]
print(nomes[1])  # Saída: Bruno
```

1.2 Operações comuns com listas

Operação	Descrição	Exemplo
Acessar índice	Recupera elemento por índice	`nomes[0]` → "Ana"
Alterar valor	Modifica valor em um índice	`nomes[1] = "Breno"`
Adicionar item	Adiciona no fim da lista	`nomes.append("Daniel")`
Inserir em índice	Adiciona em posição específica	`nomes.insert(1, "Lívia")`
Remover por valor	Remove a primeira ocorrência	`nomes.remove("Ana")`
Remover por índice	Remove item e retorna	`nomes.pop(0)`
Tamanho da lista	Retorna o número de elementos	`len(nomes)`

Exemplo combinado:

```
nomes = ["Ana", "Bruno"]
nomes.append("Carlos")
nomes[0] = "Aline"
print(nomes)  # Saída: ['Aline', 'Bruno', 'Carlos']
```

1.3 Iteração sobre listas

Exemplo simples:

```
for nome in nomes:
    print(f"Olá, {nome}!")
```

Exemplo com índices:

```
for i in range(len(nomes)):
    print(f"Posição {i}: {nomes[i]}")
```

2. Tuplas

2.1 O que são tuplas?

- Estrutura **imutável**: uma vez criada, seus valores não podem ser alterados.
- Mais rápida e segura que listas em muitos casos.
- Criadas com parênteses `()`.

Exemplo:

```
cores = ("vermelho", "verde", "azul")
print(cores[2])  # azul
```

2.2 Diferença entre lista e tupla

Característica	Lista (`[]`)	Tupla (`()`)
Mutável	Sim	Não
Uso comum	Dados dinâmicos	Dados fixos
Desempenho	Menor	Maior

Exemplo:

```
dias_semana = ("Segunda", "Terça", "Quarta")
# dias_semana[0] = "Domingo"  # ERRO! Tuplas não podem ser alteradas
```

3. DICIONÁRIOS

3.1 O que são dicionários?

- Estrutura de dados baseada em **pares chave:valor**.
- Útil para representar registros (como cadastro de um aluno ou produto).
- Criados com chaves {}.

Exemplo:

```
aluno = {
    "nome": "Maria",
    "idade": 22,
    "curso": "Python"
}
print(aluno["curso"])  # Saída: Python
```

3.2 Operações com dicionários

Ação	Código	Resultado
Acessar valor	`aluno["nome"]`	"Maria"
Adicionar chave	`aluno["nota"] = 9.5`	Chave "nota" adicionada ao dicionário
Atualizar valor	`aluno["idade"] = 23`	Altera a idade para 23
Remover chave	`del aluno["curso"]`	Remove a chave "curso"

Exemplo:

```
aluno["nota"] = 9.5
aluno["curso"] = "Python Avançado"
del aluno["idade"]
print(aluno)
```

3.3 Iterando sobre dicionários

Exemplo completo:

```
for chave, valor in aluno.items():
    print(f"{chave}: {valor}")
```

4. COMPARATIVO ENTRE LISTAS, TUPLAS E DICIONÁRIOS

Estrutura	Mutável?	Indexado?	Usa chaves?	Ideal para
Lista	Sim	Sim	Não	Coleções modificáveis ordenadas
Tupla	Não	Sim	Não	Conjuntos fixos e imutáveis
Dicionário	Sim	Não	Sim	Mapeamento e armazenamento de dados

5. Exercício Final (para resolução em sala)

Tarefa (Dificuldade Média):

Crie um sistema de cadastro de produtos com as seguintes etapas:

1. Solicite via `input()` o **nome** e o **preço** de 3 produtos.
2. Armazene os dados em um **dicionário**, onde a chave é o nome e o valor é o preço.
3. Use uma **lista** para manter a ordem de cadastro dos produtos.
4. Após o cadastro, exiba todos os produtos e seus respectivos preços com formatação de moeda.

Dica: Utilize `append()`, `f-string`, `for` e `items()`.

```python
# ------------------------------
# SISTEMA DE CADASTRO DE PRODUTOS
# ------------------------------

# Cria um dicionário vazio para armazenar produtos.
# Cada produto será armazenado como: {"nome": preço}
produtos = {}  # Dicionário → estrutura de chave:valor (ex: "Livro": 39.90)

# Cria uma lista vazia para registrar a ordem de cadastro dos nomes.
nomes_cadastrados = []  # Lista → estrutura que armazena múltiplos valores

# Mensagem de abertura do programa
print("=== Cadastro de Produtos ===")  # print() → exibe mensagens na tela

# Início de um laço de repetição que vai executar 3 vezes (de 0 até 2)
for i in range(3):  # range(3) → gera os valores 0, 1, 2
    print(f"\nProduto {i+1}")  # f-string → permite inserir variáveis dentro do texto

    # Solicita o nome do produto ao usuário e armazena na variável 'nome'
    nome = input("Digite o nome do produto: ")
    # input() → lê uma entrada digitada no teclado (sempre como string)

    # Solicita o preço do produto, converte para float e armazena na variável 'preco'
    preco = float(input("Digite o preço do produto (ex: 19.90): R$ "))
    # float() → converte texto (string) para número decimal

    # Adiciona o par nome:preço ao dicionário
    produtos[nome] = preco  # Exemplo: produtos["Livro"] = 39.90

    # Adiciona o nome do produto à lista para manter a ordem de cadastro
    nomes_cadastrados.append(nome)
    # append() → adiciona um item no final da lista

# Após o laço, exibe todos os produtos cadastrados
print("\n=== Produtos Cadastrados ===")

# Laço para percorrer os nomes na ordem que foram cadastrados
for nome in nomes_cadastrados:
    preco = produtos[nome]  # Acessa o preço correspondente no dicionário
    print(f"- {nome}: R$ {preco:.2f}")
    # :.2f → formata o número com 2 casas decimais (ex: 39.90)
```