PROGRAMAÇÃO 101

DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG

Dia 4

Tuplas

Definição e Importância

- **Definição:**
  - Tuplas é um conjunto de elementos imutável, o que a difere de vetores. É definido por meio de parênteses.

```
dados = ("Nome", "João")
coordenadas = (10.5, 20.3)
```

Definição e Importância

- Acessar determinado valor dessa tupla:

```
print(coordenadas[1])
```

- Atribuindo valores da tupla em variáveis

```
x, y = coordenadas
```

- Concatenando duas tuplas

```
tuplas_concatenadas = dados + coordenadas
```

```
('Nome', 'João', 10.5, 20.3)
```

Dia 4

Dicionários

Definição e Importância

- **Definição:**
  - Dicionários ou dicts são estruturas que armazenam chaves e valores. As chaves se encontram à esquerda dos dois pontos, enquanto os valores se encontram à direita. Usa-se chaves para criar o dict.

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

Definição e Importância

- Acessar valor desse dicionário:

```
print(dados_pessoais['nome'])
```

- Método get() coleta o valor da chave

```
print(dados_pessoais.get('nome'))
print(dados_pessoais.get('nome', 'Não Encontrado!'))
```

Definição e Importância

- Altera valor de chave:

```
dados_pessoais['nome'] = "Jorgito"
```

- Cria nova chave e atribui um valor

```
dados_pessoais['profissao'] = "Desenvolvedor"
```

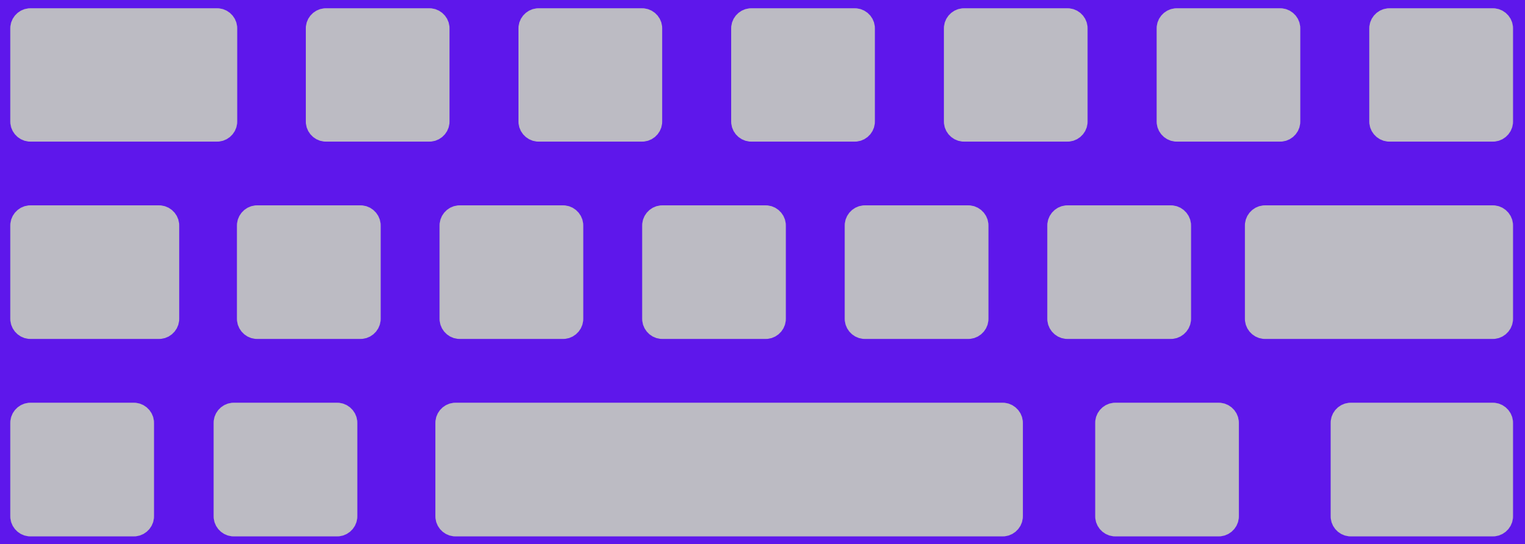
Definição e Importância

- Funções úteis:
  - **keys()**: Retorna todas as chaves do dicionário.
  - **values()**: Retorna todos os valores do dicionário.
  - **items()**: Retorna pares chave-valor como tuplas.
  - **update()**: Atualiza um dicionário com outro dicionário ou com pares chave-valor.

Dia 4

Matrizes

O que são Matrizes em Python?



Definição e Importância

- **Definição:**
  - Em Python, matrizes podem ser representadas como listas de listas, onde cada lista interna é uma linha da matriz.

```
matriz = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
```

Acesso a Elementos em Matrizes

- **Utilização de Índices Duplos:**
  - Acesso a elementos especificando a linha e a coluna.

```
elemento = matriz[1][2]
```

Manipulação de Matrizes

- **Operações Comuns:**
  - Adição de linhas ou colunas.
  - Transposição da matriz.

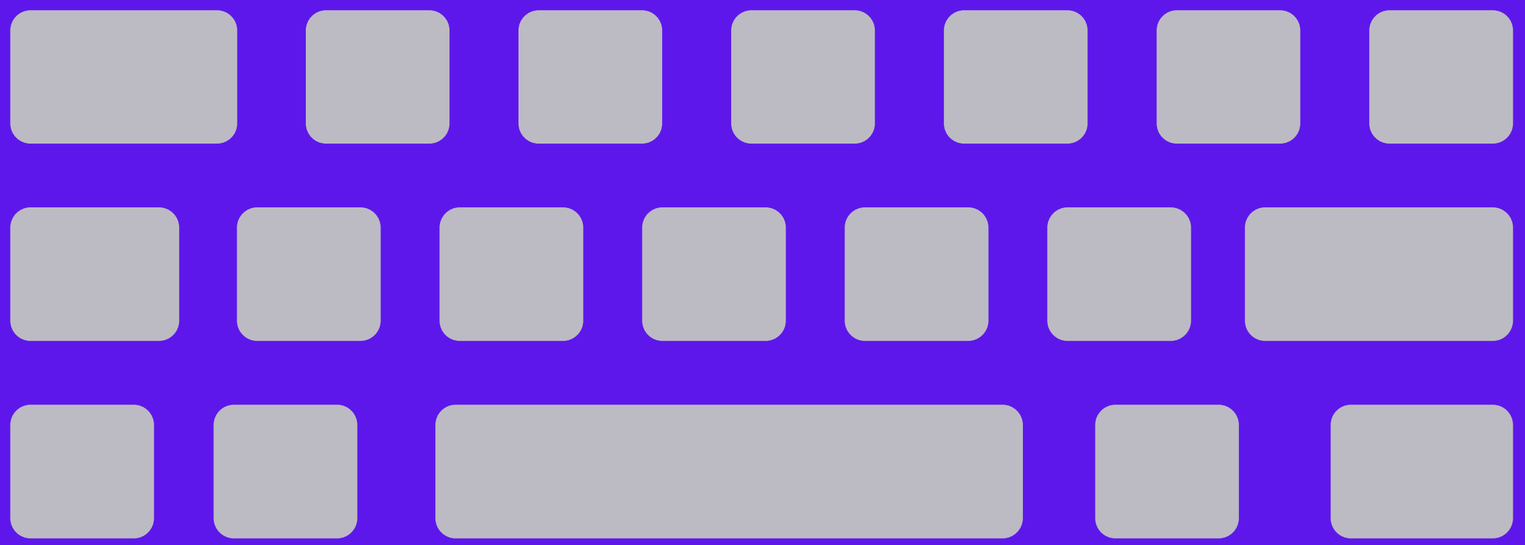
```python
# Adição de Linha
nova_linha = [10, 11, 12]
matriz.append(nova_linha)

# Adição de Coluna
for linha in matriz:
    linha.append(0)
```

Dia 4

Bibliotecas

O que são Bibliotecas em Python?



Definição e Importância

Funções evitam repetições desnecessárias!

Sem função:

```
print("\nParabéns pra você!")
print("Nesta data querida!")
print("Muitas felicidades!")
print("Muitos anos de vida!")
print("\nParabéns pra você!")
print("Nesta data querida!")
print("Muitas felicidades!")
print("Muitos anos de vida!")
print("\nParabéns pra você!")
print("Nesta data querida!")
print("Muitas felicidades!")
print("Muitos anos de vida!")
```

Com função:

```
def parabens():
    print("\nParabens pra voce!")
    print("Nesta data querida!")
    print("Muitas felicidades!")
    print("Muitos anos de vida!")


parabens()
parabens()
parabens()
```

Definição e Importância

- **Definição:**
  - Bibliotecas em Python são coleções que oferecem funcionalidades prontas. Elas facilitam o desenvolvimento ao reutilizar código pré-escrito. Ajudam em tarefas das mais básicas até extremamente complexas como I.A

Como usar?

- A maioria das bibliotecas é utilizada fazendo um “import”, seguido de qual biblioteca queremos importar. Lembrando que só poderá ser usado os conteúdos dela após a declaração.

```
import os
import json
from tqdm import tqdm
```

Dia 4

Funções Pré-Implementadas

Matrizes e NumPy

- **Biblioteca NumPy:**
  - Facilita operações matriciais e álgebra linear.

```
import numpy as np


matriz_numpy = np.array(matriz)
resultado = np.dot(matriz_numpy, matriz_numpy)
```

Como usar?

- **Definição:**
  - Tais funções, ja foram implementadas previamente e você pode utiliza-las em seu código, para resolver o que for necessário. Algumas funções requerem bibliotecas para usar, já outras ja existem no próprio python.

Como usar?

- Uma primeira função extremamente importante é a:
.sort()

```
# Definindo o vetor
vetor = [34, 12, 5, 66, 1]


# Ordenando o vetor
vetor.sort()
```

- Resultado: [1, 5, 12, 34, 66]

Métodos de Manipulação

- **upper() e lower():**
  - Transformação para maiúsculas ou minúsculas.
- **split():**
  - Divisão de uma string em uma lista de substrings.
- **join():**
  - Concatenação de elementos de uma lista em uma string.

```
# Exemplo de upper() e lower()
maiusculas = mensagem.upper()
minusculas = mensagem.lower()

# Exemplo de split()
palavras = mensagem.split(",")  # Resultado: ['Olá', ' Python!']

# Exemplo de join()
lista_palavras = ['Olá', 'Mundo']
mensagem_nova = '-'.join(lista_palavras)  # Resultado: 'Olá-Mundo'
```

Mais funções

- **min()**
  - Retorna o menor valor de um vetor
- **max()**
  - Retorna o maior valor de um vetor

```
x = min(5, 10, 25)
y = max(5, 10, 25)
```

- **abs()**
  - Retorna o valor absoluto de um número

```
x = abs(-7.25)
```

Utilizando import math

- **math.sqrt()**
  - Descobre a raiz quadrada de um número

```
import math

x = math.sqrt(64)
```

- **math.ceil()**
  - Arredonda o número para cima

- **math.floor()**
  - Arredonda o número para baixo

```
import math

x = math.ceil(1.4)
y = math.floor(1.4)

print(x) # returns 2
print(y) # returns 1
```

Utilizando import math

- **math.pi()**
  - Atribui o valor de pi para a variavel

```
import math


x = math.pi
```

- **math.factorial()**
  - Diz o fatorial de um número

- **math.log()**
  - Diz o logaritmo de um número na base e

```
import math

print(math.factorial(5))

print(math.log(10))
```

PROGRAMAÇÃO 101

DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG