PROGRAMAÇÃO 101

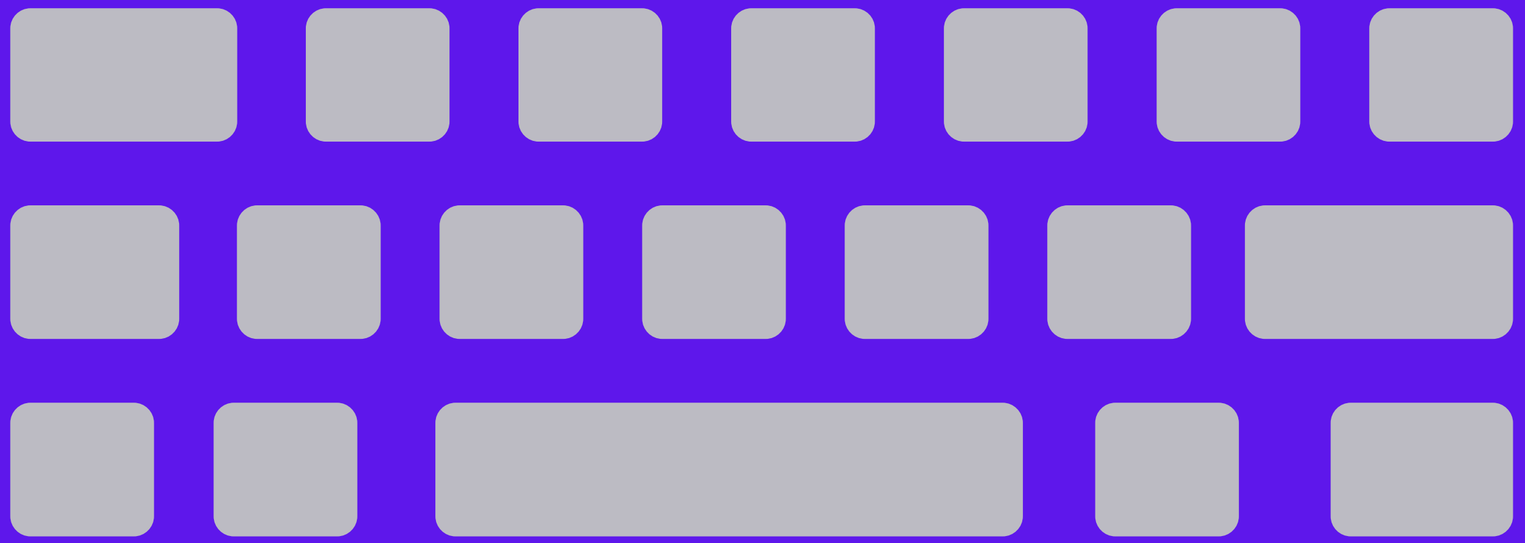
DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG

Dia 2

Comandos Condicionais

O que são Comandos Condicionais?



Definição e Importância

- **Definição:**
  - Comandos condicionais permitem que um programa execute diferentes blocos de código com base em condições específicas.
- **Necessidade:**
  - Lidar com situações em que a execução de código depende de circunstâncias variáveis.

Estrutura Básica

- **Utilização de if, elif e else:**
  - if: Executa um bloco de código se uma condição for verdadeira.
  - elif: Adiciona condições adicionais, se necessário.
  - else: Executa um bloco de código se nenhuma condição anterior for verdadeira.

Operadores Lógicos

- **and, or, not:**
  - Permitem combinar múltiplas condições lógicas.
  - and: Ambas as condições devem ser verdadeiras.
  - or: Pelo menos uma condição deve ser verdadeira.
  - not: Inverte o valor de verdade de uma condição.

- Utilização do if, elif e else:
  - Permite a execução condicional de blocos de código.

```
idade = 18

if idade < 18:
    print("Menor de idade")
elif idade >= 18 and idade < 65:
    print("Adulto")
else:
    print("Idoso")
```

- Operadores and, or, not:
  - Permitem combinar condições lógicas.

```
salario = 3000
horas_trabalhadas = 40

if salario > 2500 and horas_trabalhadas >= 40:
    print("Bônus por horas extras!")
```

- Múltiplos Níveis de if:
  - Aninhar condicionais para verificar várias condições.

```python
temperatura = 25

if temperatura > 30:
    print("Está quente!")
else:
    if temperatura > 20:
        print("Agradável")
    else:
        print("Está frio!")
```

Expressões Condicionais (Operador Ternário)

Forma Concisa de if-else:

- Permite uma atribuição com base em uma condição em uma única linha.
- Sintaxe: valor_verdadeiro if condicao else valor_falso.

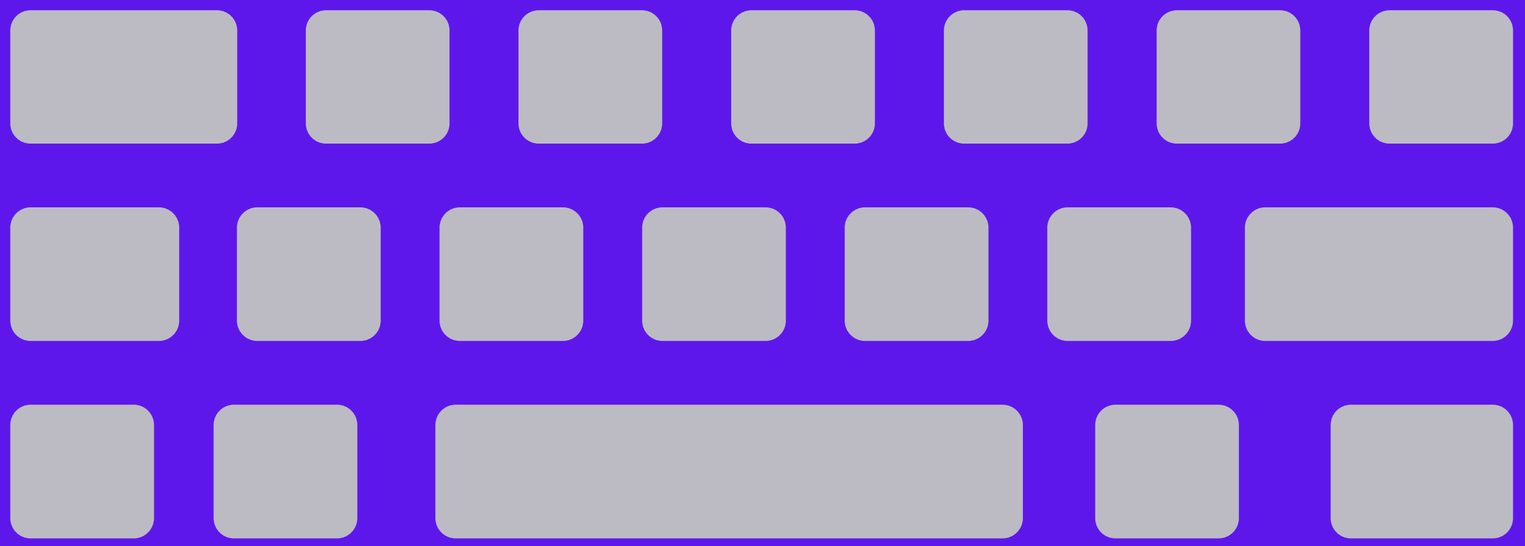
- Forma Concisa de if-else:
  - Permite uma atribuição com base em uma condição.

```
idade = 22
status = "Maior de idade" if idade >= 18 else "Menor de idade"
```

Dia 2

Vetores

O que são Vetores em Python?



Definição e Importância

- **Definição:**
  - Em Python, vetores são representados por listas, que são coleções ordenadas e mutáveis de elementos.

```
numeros = [1, 2, 3, 4, 5]
```

Manipulação de Vetores

- **Acesso a Elementos:**
  - Utilização de índices para acessar elementos.
  - Índices começam do zero.
- **Operações Comuns:**
  - Adição de elementos: vetor.append(elemento).
  - Remoção de elementos: vetor.remove(elemento).

```
# Exemplo de Acesso a Elementos
primeiro_elemento = numeros[0]

# Exemplo de Adição de Elementos
numeros.append(6)

# Exemplo de Remoção de Elementos
numeros.remove(3)
```

Operações Matemáticas com Vetores

- **Soma de Vetores:**
  - Adição elemento a elemento.
- **Multiplicação por um número:**
  - Multiplicação de cada elemento do vetor por um número.
- **Concatenação de Vetores:**
  - Junção de duas sequências sem modificar seus elementos.

```
vetor1 = [1, 2, 3]
vetor2 = [4, 5, 6]

vetor_concat = vetor1 + vetor2 # [1, 2, 3, 4, 5, 6]
```

O que são Strings ?

- Definição:
  - Strings são sequências imutáveis de caracteres em Python.

```
mensagem = "Olá, Python!"
```

Manipulação de Strings

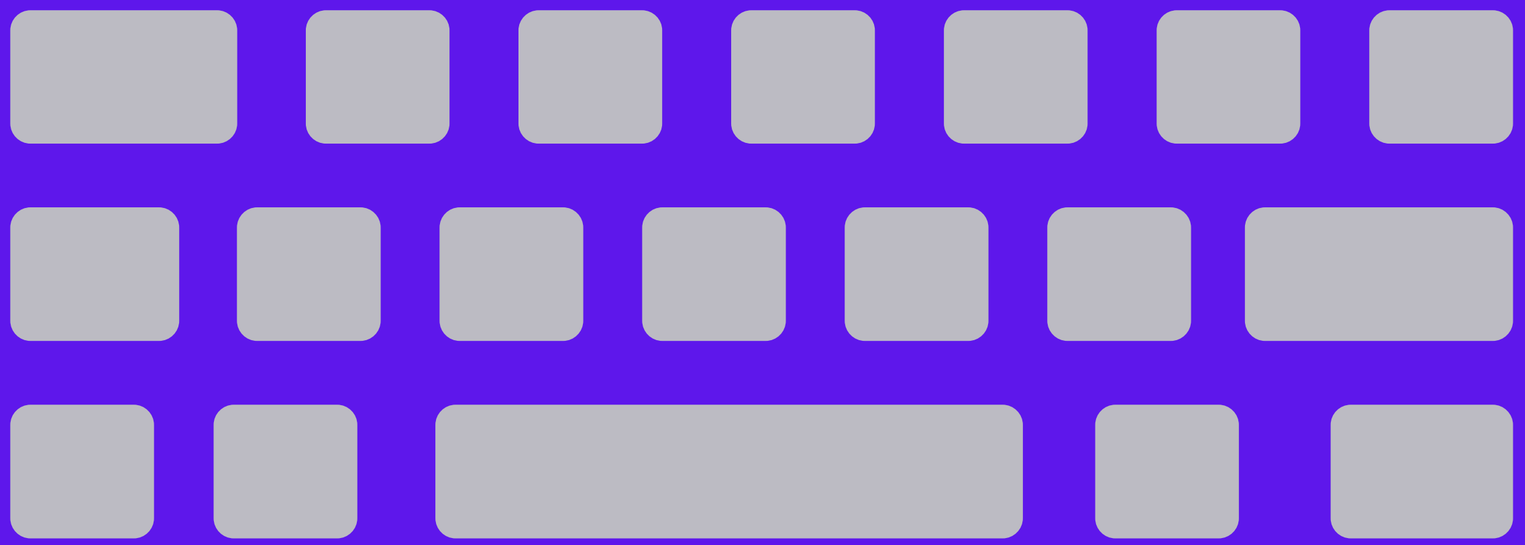
- **Concatenação:**
  - Combinação de strings utilizando o operador +.
- **Comprimento da String:**
  - Determinado pelo método len().
- **Índices e Fatiamento:**
  - Acesso a caracteres específicos e fatiamento de strings.

```python
# Exemplo de Concatenacao
saudacao = "Ola, " + "Mundo!"

# Exemplo de Comprimento da String
mensagem = "batata"
tamanho = len(mensagem) # tamanho = 6

# Exemplo de Indices e Fatiamento
primeiro_caracter = mensagem [0] # 'b'
fatia = mensagem [4:6] # fatia = 'ta'
```

Dia 2

Boas Práticas

Boas Práticas de Programação em Python



Legibilidade e Clareza

Nomes de Variáveis Descritivos:

- Escolha nomes de variáveis que descrevam claramente sua finalidade.
- Prefira nomes legíveis em vez de abreviações obscuras.

```
# Ruim
a = 10
# Bom
numero_de_estudantes = 10
```

Comentários e Documentação

Comentários e documentação são elementos cruciais para comunicar a lógica do código e facilitar a compreensão para outros desenvolvedores ou para você mesmo no futuro.

Comentários para Explicar Lógica Complexa:

```
# Ruim - Lógica obscura
resultado = a * b + c / d

# Bom - Comentário explicativo
# Calcula o resultado usando a fórmula a * b + c / d
resultado = a * b + c / d
```

Evite Códigos Duplicados

A duplicação de código pode levar a problemas de manutenção, aumentar a propensão a erros e dificultar a atualização do software.

Reutilização de variáveis

```python
# Ruim
numero = 7
print(numero)
numero = 8
print(numero)

# Ideal
primeiro_numero = 7
print(primeiro_numero)
segundo_numero = 8
print(segundo_numero)
```

Identação

Identação é o espaçamento que inserimos antes de uma linha de código. É considerada uma boa prática pois melhora a legibilidade do código. Em Python, essa prática é obrigatória.

Para realizar a identação, usa-se a tabulação (**tab**)

```python
# Sem identação
tamanho = 10
if tamanho > 10:
print("O tamanho é maior")
else:
print("O tamanho é menor")

# Com identação
tamanho = 10
if tamanho > 10:
    print("O tamanho é maior")
else:
    print("O tamanho é menor")
```

PROGRAMAÇÃO 101

DCC
DEPARTAMENTO DE
CIÊNCIA DA COMPUTAÇÃO

UFMG