**Matéria Adiantada dia 04/11**

---

## Semana 30 - Aula 1

Tópico Principal da Aula: Estruturas de Decisão Composta

Subtítulo/Tema Específico: Aninhamento de Estruturas de Decisão (IF Aninhado)

Código da aula: [SIS]ANO1C1B4S30A1

**Objetivos da Aula:**

- Compreender o conceito de aninhamento (nested) de comandos condicionais na lógica de programação.
- Desenvolver programas que exijam múltiplos níveis de avaliação de condições para determinar o fluxo de execução.
- Aplicar estruturas if-elif-else de forma aninhada na sintaxe Python.

**Recursos Adicionais (Sugestão, pode ser adaptado):**

- Caderno para anotações;
- Acesso ao laboratório de informática e/ou internet.

**Exposição do Conteúdo:**

**Referência do Slide:** Slide 06 - Estrutura Condicional Aninhada

- **Definição:** A estrutura condicional aninhada (ou *nested if*) ocorre quando uma ou mais estruturas de decisão (if, elif ou else) estão contidas dentro do bloco de código de outra estrutura de decisão. Ela é utilizada quando o teste da segunda condição só faz sentido ou é necessário após a primeira condição ser avaliada como verdadeira ou falsa.
- **Aprofundamento/Complemento (se necessário):** Em Python, o aninhamento é definido pela **indentação**. O uso excessivo de *ifs* aninhados (muitos níveis) é geralmente considerado uma má prática, pois torna o código difícil de ler, manter e depurar. Em muitos casos, o aninhamento pode ser simplificado usando operadores lógicos (and, or, not) – tema da próxima aula – ou utilizando a estrutura if-elif-else sequencial para condições mutuamente exclusivas.
- **Exemplo Prático:** Avaliar a elegibilidade de um aluno para uma bolsa.
- Python

```
nota = 8.5
renda_familiar = 1500

if nota >= 7.0:
    if renda_familiar <= 2000:
        print("Aluno elegível para bolsa.")
    else:
```

```
        print("Nota suficiente, mas renda excede o limite.")
else:
    print("Nota insuficiente para elegibilidade inicial.")
```

- 
    - **Link de Vídeo 1:** 15 - Python - Nested Conditional Decision Structure - IF..THEN..ELSE IF - YouTube
    - **Link de Vídeo 2:** Curso Python #012 - Condições Aninhadas - YouTube

---

## Semana 30 - Aula 2

Tópico Principal da Aula: Estruturas de Decisão Composta

Subtítulo/Tema Específico: Operadores Lógicos (and, or, not)

Código da aula: [SIS]ANO1C1B4S30A2

**Objetivos da Aula:**

- Conhecer a função e a precedência dos operadores lógicos (Booleano) `and`, `or` e `not` em Python.
- Aprender a combinar múltiplas expressões de teste em uma única condição.
- Escrever código mais limpo e conciso, evitando aninhamentos desnecessários.

**Recursos Adicionais (Sugestão, pode ser adaptado):**

- Caderno para anotações;
- Acesso ao laboratório de informática e/ou internet.

**Exposição do Conteúdo:**

**Referência do Slide:** Slide 06 - Introdução aos Operadores Lógicos

- **Definição:** Operadores lógicos são utilizados para combinar expressões condicionais, retornando um valor Booleano (`True` ou `False`). Eles permitem testar múltiplas condições simultaneamente na mesma linha de código.
- **Aprofundamento/Complemento (se necessário):**
    - **`and` (E):** Retorna `True` se **todas** as condições combinadas forem verdadeiras. É o operador mais restritivo.
    - **`or` (OU):** Retorna `True` se **pelo menos uma** das condições combinadas for verdadeira.
    - **`not` (NÃO):** Inverte o valor lógico de uma expressão (transforma `True` em `False` e vice-versa).
    - **Precedência:** Em expressões complexas, o operador `not` tem a maior precedência, seguido pelo `and`, e por último o `or`. O uso de parênteses é crucial para garantir que a lógica seja avaliada na ordem desejada.
- **Exemplo Prático:** Verificar se um usuário pode acessar um sistema (deve ser administrador **E** estar ativo).
- Python

```
nivel_acesso = "admin"
status_ativo = True

if nivel_acesso == "admin" and status_ativo == True:
    print("Acesso concedido: Administrador Ativo.")
else:
    print("Acesso negado.")
```

- 
    - **Link de Vídeo 1:** Python Tutorial #16: A closer look at logical operators and, or, not - YouTube
    - **Link de Vídeo 2:** Logical Operators in Python (AND, OR, NOT) | Python Tutorial for Beginners - YouTube

---

## Semana 30 - Aula 3

Tópico Principal da Aula: Criação de Programas: Prática

Subtítulo/Tema Específico: Uso de if-elif-else em Casos Complexos e Introdução a Estruturas de Dados

Código da aula: [SIS]ANO1C1B4S30A3

**Objetivos da Aula:**

- Dominar a utilização da sequência `if-elif-else` para criar um fluxo de decisão eficiente e mútuo-exclusivo.
- Aplicar estruturas de decisão para processar dados armazenados em vetores (listas) em Python.
- Desenvolver a lógica para problemas cotidianos que exigem a classificação de valores.

**Recursos Adicionais (Sugestão, pode ser adaptado):**

- Caderno para anotações;
- Acesso ao laboratório de informática e/ou internet.

**Exposição do Conteúdo:**

**Referência do Slide:** Slide 06 - Sequência `if-elif-else`

- **Definição:** A sequência `if-elif-else` (onde `elif` significa *else if*) é a forma ideal em Python de lidar com múltiplos caminhos de decisão que são **mutuamente exclusivos**. A execução pára no primeiro teste que resultar em verdadeiro, tornando o código mais rápido e mais limpo do que vários `if` separados.
- **Aprofundamento/Complemento (se necessário):** Enquanto estruturas aninhadas (`if` dentro de `if`) são usadas para dependência lógica (só testa B se A for verdadeiro), o `if-elif-else` é usado para exclusividade (se A for verdadeiro, ignora B, C e D). A prática da aula 3 foca em como utilizar essa estrutura de decisão em conjunto com a

leitura ou manipulação de coleções de dados (como listas/vetores) para resolver problemas que envolvem múltiplos resultados, como a classificação de notas ou faixas etárias.

- **Exemplo Prático:** Classificação de Notas.
- Python

```
nota = 75

if nota >= 90:
    conceito = "A"
elif nota >= 80:
    conceito = "B"
elif nota >= 70:
    conceito = "C"
else:
    conceito = "D"

print(f"O aluno obteve o conceito: {conceito}")
```

- 
    - **Link de Vídeo 1:** Python 3 Programming Tutorial: If Elif Else - YouTube
    - **Link de Vídeo 2:** The if-elif-else Statement in Python - YouTube

---

## Semana 30 - Aula 4

Tópico Principal da Aula: Estruturas de Decisão Composta

Subtítulo/Tema Específico: Operador Condicional Ternário

Código da aula: [SIS]ANO1C1B4S30A4

**Objetivos da Aula:**

- Compreender o conceito e a sintaxe do operador ternário em Python.
- Aplicar o operador ternário para escrever estruturas `if-else` simples em uma única linha, visando concisão.
- Identificar situações onde o uso do operador ternário é apropriado para aumentar a legibilidade do código.

**Recursos Adicionais (Sugestão, pode ser adaptado):**

- Caderno para anotações;
- Acesso ao laboratório de informática e/ou internet.

**Exposição do Conteúdo:**

**Referência do Slide:** Slide 06 - Operador Condicional Ternário

- **Definição:** O operador ternário em Python (ou `if-else` em uma linha) é uma expressão condicional que permite atribuir um valor a uma variável com base em uma condição, tudo em uma única linha de código. É chamado de "ternário" porque envolve três operandos: o valor se verdadeiro, a condição e o valor se falso.
- **Aprofundamento/Complemento (se necessário):** A sintaxe é: `resultado = valor_se_verdadeiro if condicao else valor_se_falso`. Este operador é ideal para situações simples de atribuição e deve ser evitado para lógica complexa, pois pode prejudicar a legibilidade. Seu principal objetivo é a concisão, substituindo a estrutura tradicional de 4 linhas por apenas 1.
- **Exemplo Prático:** Determinar o status de votação.
- Python

```
idade = 19
status_votacao = "Obrigatório" if idade >= 18 else "Não obrigatório"

print(f"Status: {status_votacao}")

# Forma tradicional (equivalente)
# if idade >= 18:
#     status_votacao = "Obrigatório"
# else:
#     status_votacao = "Não obrigatório"
```

-
    - **Link de Vídeo 1:** Python If-Else Statement in One Line - Ternary Operator Explained | Better Data Science
    - **Link de Vídeo 2:** Operador ternário no Python (if ternário) - Alonza