Semana 17 - Aula 1

Tópico Principal da Aula: Estruturas de Decisão Composta em Python

Subtítulo/Tema Específico: Introdução às Estruturas de Decisão Composta

Código da aula: [SIS]ANO1C1B3S17A1

Objetivos da Aula:

- Compreender os conceitos sobre o desenvolvimento e a execução prática de programas computacionais utilizando estruturas de decisão compostas.[1]
- Desenvolver sistemas computacionais utilizando ambiente de desenvolvimento.[2]
- Migrar sistemas implementando rotinas e estruturas de dados mais eficazes.[3]
- Resolver problemas computacionais com estratégias criativas.[4]

Recursos Adicionais (Sugestão, pode ser adaptado):

- Caderno para anotações[5];
- Acesso ao laboratório de informática e/ou internet[6];
- Recurso audiovisual para exibição de vídeos e imagens[7].

Exposição do Conteúdo:

Referência do Slide: Slides 06 e 07 - Estruturas de Decisão Compostas

- **Definição:** Uma estrutura de decisão composta é um componente fundamental na lógica de programação que permite a um programa avaliar uma ou mais condições e, com base no resultado dessa avaliação (verdadeiro ou falso), direcionar o fluxo de execução para diferentes blocos de código. Isso confere ao programa a capacidade de responder de maneira dinâmica a diferentes entradas ou situações, tornando-o mais inteligente e adaptável.
- **Aprofundamento/Complemento (se necessário):** As condições avaliadas são geralmente expressões booleanas que resultam em `true` (verdadeiro) ou `false` (falso). Em Python, a indentação (espaços ou tabulações no início da linha) é crucial para definir os blocos de código associados a cada parte da estrutura de decisão (if, elif, else). Um erro comum é a indentação incorreta, que pode levar a comportamentos inesperados do programa. Embora Python permita aninhar estruturas de decisão (colocar uma dentro da outra), é importante usar essa funcionalidade com moderação, pois o aninhamento excessivo pode dificultar a leitura e a manutenção do código.
- **Exemplo Prático:** Imagine um sistema de semáforo. Se a luz for verde, o carro pode seguir. Se for amarela, o carro deve diminuir a velocidade. Se for vermelha, o carro deve parar.
- Python

```
cor_semaforo = "verde"
```

```
if cor_semaforo == "verde":
    print("Siga em frente!")
elif cor_semaforo == "amarelo":
    print("Atenção! Diminua a velocidade.")
else: # Cor vermelha
    print("Pare!")
```

Referência do Slide: Slide 08 - Principais Estruturas em Python: If, Else, Elif

- **Definição:** As principais estruturas de decisão compostas em Python são `if`, `else` e `elif`.
    - **If:** É a forma mais básica. O bloco de código sob o `if` será executado somente se a condição especificada for verdadeira (true). 14
    - **Else:** Utilizado em conjunto com o `if`. Se a condição do `if` não for verdadeira, o bloco de código sob o `else` será executado. O
    - `else` é opcional; pode-se ter um `if` sem um `else` se não houver necessidade de executar uma ação para a condição falsa.
    - **Elif (else if):** Permite verificar múltiplas condições em sequência. Se a condição no `if` for falsa, o programa verifica as condições nos blocos `elif` subsequentes. 1
- **Aprofundamento/Complemento (se necessário):** A ordem das condições `if`, `elif`, `else` é crucial. O Python avalia as condições de cima para baixo e executa o bloco de código da *primeira* condição que for verdadeira. Uma vez que um bloco é executado, o restante da estrutura de decisão é ignorado. Isso significa que, mesmo que múltiplas condições sejam verdadeiras, apenas a primeira encontrada será processada.
- **Exemplo Prático:**
- Python

```
nota = 75

if nota >= 90:
    print("Excelente!")
elif nota >= 70:
    print("Bom!")
elif nota >= 50:
    print("Regular.")
else:
    print("Insatisfatório.")
```

-
-
- **Vídeos Sugeridos:**
    - Programação Web com Python e Django. Estrutura de Decisão Simples e Composta (IF ELSE ELIF). Link: https://www.youtube.com/watch?v=kYJv9c79yX0
    - TreinaWeb. Estruturas Condicionais no Python (if, else, elif). Link: https://www.youtube.com/watch?v=s_pW2sR1nug

Referência do Slide: Slide 09 - Exemplo de Indentação e Estrutura `if-elif-else`

- **Definição:** O slide apresenta um exemplo prático de como as estruturas `if`, `elif` e `else` são usadas em Python, enfatizando a importância da indentação para definir os blocos de código.[18]
- **Aprofundamento/Complemento (se necessário):** A indentação em Python não é meramente uma questão de estilo; é uma parte fundamental da sintaxe da linguagem que define a hierarquia e o escopo dos blocos de código. Blocos de código com a mesma indentação são considerados parte do mesmo nível de controle. Uma indentação inconsistente pode gerar erros de sintaxe ou, pior, lógica errada que não é imediatamente aparente.
- **Exemplo Prático:** (Exemplo do slide com explicação)
- Python

```
idade = 18

if idade < 13: # Se a idade for menor que 13
    print("Criança") # Este bloco será executado
elif idade < 18: # Se a condição acima for falsa E a idade for menor que 18
    print("Adolescente") # Este bloco será executado
else: # Se NENHUMA das condições acima for verdadeira (idade maior ou igual a 18)
    print("Adulto") # Este bloco será executado
```

-
- Neste exemplo, a variável `idade` é `18`. A primeira condição (`idade < 13`) é falsa. A segunda condição (`idade < 18`) também é falsa. Portanto, o bloco `else` é executado, e o programa imprime "Adulto".
- **Vídeos Sugeridos:**
  - Boson Treinamentos. Python - if, elif e else - Estruturas Condicionais (Aula 11). Link: https://www.youtube.com/watch?v=1F2b_k_w5zI
  - Curso em Vídeo. Curso Python #08 - Condições Aninhadas. Link: https://www.youtube.com/watch?v=t5Jv1Tq_r-Y

Semana 17 - Aula 2

Tópico Principal da Aula: Aplicação de Estruturas de Decisão Composta

Subtítulo/Tema Específico: Funções em Python e Sistema de Classificação

Código da aula: [SIS]ANO1C1B3S17A2

Objetivos da Aula:

- Compreender os conceitos sobre o desenvolvimento e a execução prática de programas computacionais utilizando estruturas de decisão compostas.[19]
- Desenvolver sistemas computacionais utilizando ambiente de desenvolvimento.[20]
- Migrar sistemas implementando rotinas e estruturas de dados mais eficazes.[21]
- Resolver problemas computacionais com estratégias criativas.[22]

Recursos Adicionais (Sugestão, pode ser adaptado):

- Caderno para anotações[23];
- Acesso ao laboratório de informática e/ou internet[24];
- Recurso audiovisual para exibição de vídeos e imagens[25].

Exposição do Conteúdo:

Referência do Slide: Slide 06 - Funções em Python

- **Definição:** O conceito de funções em Python é introduzido para auxiliar na organização e modularização do código, especialmente ao lidar com decisões compostas.[26] Funções são blocos de código reutilizáveis que realizam uma tarefa específica. Elas ajudam a dividir programas complexos em partes menores e mais gerenciáveis, melhorando a legibilidade e a manutenção.
- **Aprofundamento/Complemento (se necessário):** Funções podem aceitar parâmetros (entradas) e retornar valores (saídas). Ao encapsular lógicas de decisão dentro de funções, o código se torna mais limpo e evita repetições. Por exemplo, uma função pode receber uma idade e retornar a classificação etária, usando estruturas `if-elif-else` internamente.
- **Exemplo Prático:**
- Python

```
def saudar(nome):
    """Esta função saúda a pessoa com o nome fornecido."""
    print(f"Olá, {nome}!")

saudar("Ana") # Chama a função, imprimindo "Olá, Ana!"
saudar("Pedro") # Chama a função, imprimindo "Olá, Pedro!"
```

-
-
- **Vídeos Sugeridos:**
    - Alura. PYTHON para Data Science: trabalhando com funções, estruturas de dados e exceções. 02 Built-in function.[27] Link:
    - https://cursos.alura.com.br/course/python-data-science-funcoes-estruturas-dados-excecoes/task/125896
    - Curso em Vídeo. Curso Python #15 - Funções (Parte 1). Link: https://www.youtube.com/watch?v=FNf1W0gWw0Q

Referência do Slide: Slides 07 e 08 - Sistema de Classificação de Filmes

- **Definição:** Este exemplo ilustra a aplicação de estruturas de decisão compostas para classificar filmes com base na faixa etária, utilizando as categorias G (geral), PG (orientação parental, para menores de 10 anos acompanhados) e R (restrito, apenas para maiores de 18 anos).[28] A resolução demonstra como o
- `elif` é usado para verificar múltiplas condições sequencialmente.[29]
- **Aprofundamento/Complemento (se necessário):** O uso de `if-elif-else` é ideal para cenários onde você tem várias condições mutuamente exclusivas, ou seja, apenas uma delas pode ser verdadeira por vez. No exemplo, um filme não pode ser "G" e

"R" ao mesmo tempo para a mesma classificação. A estrutura garante que, assim que uma condição é atendida, o bloco correspondente é executado e as demais condições são ignoradas, otimizando o processamento.

- **Exemplo Prático:** (Exemplo do slide com explicação)
- Python

```
classificacao = input("Digite a classificação do filme (G, PG, R): ")

if classificacao == "G":
    mensagem = "Apropriado para todas as idades."
elif classificacao == "PG":
    mensagem = "Crianças menores de 10 anos devem estar acompanhadas."
elif classificacao == "R":
    mensagem = "Apenas para maiores de 18 anos."
else:
    mensagem = "Classificação desconhecida."
print(mensagem)
```

-
- Neste programa, o usuário insere uma classificação. O programa verifica as condições em ordem. Se for "G", exibe a primeira mensagem. Se não for "G" mas for "PG", exibe a segunda, e assim por diante. Se nenhuma das classificações conhecidas for inserida, a mensagem de "Classificação desconhecida" será exibida.
- **Vídeos Sugeridos:**
  - PYTHONANDO. Estrutura de decisão com Python (IF ELSE ELIF) | Aula semanal #9.[30] Link:
  - https://www.youtube.com/watch?v=1brZTXo68
  - Felipe Fialho. Python: if, else, elif - Estrutura Condicional | Curso de Python 3 #05. Link: https://www.youtube.com/watch?v=kYJv9c79yX0

Semana 17 - Aula 3

Tópico Principal da Aula: Elaboração de Programas com Decisões Compostas

Subtítulo/Tema Específico: Funções e Programa de Dieta Personalizada

Código da aula: [SIS]ANO1C1B3S17A3

Objetivos da Aula:

- Compreender os conceitos sobre o desenvolvimento e a execução prática de programas computacionais utilizando estruturas de decisão compostas.[31]
- Desenvolver sistemas computacionais utilizando ambiente de desenvolvimento.[32]
- Migrar sistemas implementando rotinas e estruturas de dados mais eficazes.[33]
- Resolver problemas computacionais com estratégias criativas.[34]

Recursos Adicionais (Sugestão, pode ser adaptado):

- Caderno para anotações[35];
- Acesso ao laboratório de informática e/ou internet[36];
- Recurso audiovisual para exibição de vídeos e imagens[37].

Exposição do Conteúdo:

Referência do Slide: Slide 06 - Funções em Python (continuação)

- **Definição:** A aula continua aprofundando o desenvolvimento de funções em Python, reforçando sua importância na estruturação de programas complexos que utilizam decisões compostas.[38]
- **Aprofundamento/Complemento (se necessário):** Funções não apenas evitam a duplicação de código, mas também tornam o programa mais legível e fácil de depurar. Quando uma lógica de decisão é encapsulada em uma função, alterações futuras podem ser feitas em um único lugar, sem a necessidade de modificar várias partes do código. Isso é fundamental para a manutenção de sistemas em larga escala.
- **Exemplo Prático:**
- Python

```
def calcular_media(n1, n2, n3):
    """Calcula a média de três notas."""
    media = (n1 + n2 + n3) / 3
    return media

nota_final = calcular_media(7, 8, 9)
print(f"A média final é: {nota_final:.2f}")

if nota_final >= 7:
    print("Aprovado!")
else:
    print("Reprovado.")
```

-
-
- **Vídeos Sugeridos:**
  - Alura. PYTHON para Data Science: trabalhando com funções, estruturas de dados e exceções. 05 Criando funções.[39] Link:
  - https://cursos.alura.com.br/course/python-data-science-funcoes-estruturas-dados-excecoes/task/125897
  - Dev Aprender. Python AULA 18 - FUNÇÕES. Link: https://www.youtube.com/watch?v=pTs7B_V8FZM

Referência do Slide: Slides 07 e 08 - Programa de Dieta Personalizada

- **Definição:** Este exemplo demonstra como aplicar estruturas de decisão compostas para recomendar uma dieta personalizada com base no objetivo de saúde do usuário: "perda de peso" (dieta baixa em calorias), "ganho de massa muscular" (alimentos ricos em proteínas) ou "manter a saúde" (dieta balanceada).[40] O

- else é destacado como um "caso padrão", executado quando nenhuma das condições if ou elif anteriores é verdadeira.[41]
- **Aprofundamento/Complemento (se necessário):** O "caso padrão" do else é crucial para garantir que o programa lide com todas as entradas possíveis, mesmo aquelas não explicitamente previstas nas condições if e elif. Sem o else, se o usuário digitasse um objetivo não reconhecido, o programa não forneceria nenhuma recomendação de dieta, o que poderia ser confuso ou frustrante. O else atua como uma "captura" para todas as outras possibilidades.
- **Exemplo Prático:** (Exemplo do slide com explicação)
- Python

```
objetivo = input("Qual é o seu objetivo de saúde (perda de peso, ganho de massa, manter a saúde)? ").lower()
# Usamos .lower() para converter a entrada para minúsculas e facilitar a comparação

if objetivo == "perda de peso":
    dieta = "Dieta baixa em calorias."
elif objetivo == "ganho de massa":
    dieta = "Alimentos ricos em proteínas."
elif objetivo == "manter a saúde":
    dieta = "Dieta balanceada."
else:
    dieta = "Objetivo desconhecido. Consulte um nutricionista."
print(dieta)
```

-
- Neste programa, a entrada do usuário é convertida para minúsculas (.lower()) para garantir que a comparação seja insensível a maiúsculas/minúsculas. Dependendo do objetivo inserido, uma dieta específica é recomendada. Se o objetivo não corresponder a nenhum dos pré-definidos, uma mensagem padrão é exibida, direcionando o usuário para um nutricionista.
- **Vídeos Sugeridos:**
  - CANAL DO OVIDIO. Algoritmos e Lógica de Programação Aula 01 – Contextualização.[42] Link:
  - https://www.youtube.com/watch?v=BuTDuahtBurfbmPsIJB2NR8
  - Gustavo Guanabara. Curso em Vídeo Python #08 - Condições Aninhadas. Link: https://www.youtube.com/watch?v=t5Jv1Tq_r-Y

Semana 17 - Aula 4

Tópico Principal da Aula: Resolução de Problemas com Estruturas de Decisão Aninhadas

Subtítulo/Tema Específico: Funções com Retorno e Sistema de Controle de Estoque

Código da aula: [SIS]ANO1C1B3S17A4

Objetivos da Aula:

- Compreender os conceitos sobre o desenvolvimento e a execução prática de programas computacionais utilizando estruturas de decisão compostas.[43]

- Desenvolver sistemas computacionais utilizando ambiente de desenvolvimento.[44]
- Migrar sistemas implementando rotinas e estruturas de dados mais eficazes.[45]
- Resolver problemas computacionais com estratégias criativas.[46]

Recursos Adicionais (Sugestão, pode ser adaptado):

- Caderno para anotações[47];
- Acesso ao laboratório de informática e/ou internet[48];
- Recurso audiovisual para exibição de vídeos e imagens[49].

Exposição do Conteúdo:

Referência do Slide: Slide 06 - Funções em Python (Funções com retorno)

- **Definição:** A aula avança no estudo de funções em Python, com foco em "Funções com retorno".[50] Uma função com retorno é aquela que, após executar sua lógica, devolve um valor para o local onde foi chamada, permitindo que esse valor seja utilizado em outras partes do programa.
- **Aprofundamento/Complemento (se necessário):** O comando `return` é usado dentro de uma função para enviar um valor de volta. Isso é extremamente útil quando a função precisa calcular algo ou processar dados e o resultado precisa ser usado fora da função. Funções com retorno são essenciais para construir programas modulares e eficientes, onde diferentes partes do código se comunicam trocando valores.
- **Exemplo Prático:**
- Python

```
def verificar_aprovacao(media):
    """Verifica se um aluno foi aprovado com base na média."""
    if media >= 7:
        return "Aprovado"
    else:
        return "Reprovado"

media_aluno = 7.5
status = verificar_aprovacao(media_aluno)
print(f"O aluno está: {status}") # Saída: O aluno está: Aprovado
```

-
-
- **Vídeos Sugeridos:**
  - Alura. PYTHON para Data Science: trabalhando com funções, estruturas de dados e exceções. 08 Funções com retorno.[51] Link:
  - https://cursos.alura.com.br/course/python-data-science-funcoes-estruturas-dados-excecoes/task/125898
  - Didática Tech. Funções com retorno em Python. Link: https://www.youtube.com/watch?v=FqS7b1zM34w

Referência do Slide: Slides 07, 08 e 09 - Sistema de Controle de Estoque

- **Definição:** Este exemplo prático demonstra a criação de um sistema de controle de estoque que alerta quando um item precisa ser reabastecido, com limites diferentes para cada categoria de produto: Smartphones (abaixo de 10), Televisores (abaixo de 5), Laptops (abaixo de 8) e Outros produtos (abaixo de 15).[52] A resolução utiliza estruturas de decisão aninhadas para gerenciar as condições específicas de cada produto.[53]
- **Aprofundamento/Complemento (se necessário):** O exemplo do controle de estoque é um ótimo caso de uso para aninhamento de `if-elif-else`. Primeiro, o programa decide qual categoria de produto está sendo verificada (`if produto == "Smartphone"`). Dentro de cada categoria, há outra decisão (`if quantidade < limite`) para verificar se o estoque está abaixo do limite específico daquela categoria. Isso demonstra como combinar decisões para lidar com lógicas mais complexas e multifacetadas.
- **Exemplo Prático:** (Exemplo do slide com explicação)
- Python

```
produto = input("Digite o tipo de produto (Smartphone, Televisor, Laptop, Outro): ")
quantidade = int(input("Digite a quantidade em estoque: "))

if produto == "Smartphone":
    if quantidade < 10: # Condição aninhada para Smartphones
        print("Reabastecer estoque de smartphones.")
    else:
        print("Estoque de smartphones está adequado.")
elif produto == "Televisor":
    if quantidade < 5: # Condição aninhada para Televisores
        print("Reabastecer estoque de televisores.")
    else:
        print("Estoque de televisores está adequado.")
elif produto == "Laptop":
    if quantidade < 8: # Condição aninhada para Laptops
        print("Reabastecer estoque de laptops.")
    else:
        print("Estoque de laptops está adequado.")
else: # Para "Outro" ou qualquer produto não listado
    if quantidade < 15: # Condição aninhada para Outros produtos
        print("Reabastecer estoque de outros produtos.")
    else:
        print("Estoque de outros produtos está adequado.")
```

-
- Neste código, a primeira camada de `if-elif-else` determina a categoria do produto. Uma vez que a categoria é identificada, uma segunda camada de `if-else` é usada para verificar a quantidade em estoque em relação ao limite específico daquela categoria, fornecendo a mensagem apropriada.
- **Vídeos Sugeridos:**
    - CANAL DO OVIDIO. Algoritmos e Lógica de Programação Aula 02 – Tipos de Operadores.[54] Link:
    - https://www.youtube.com/watch?v=ARZIhDhQ
    - Didática Tech. Estruturas condicionais (if/else/elif) e operadores lógicos em Python. Link: https://www.youtube.com/watch?v=uK4jB2V4f08