Educação Profissional Paulista

Técnico em
Desenvolvimento de Sistemas

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO

Estruturas de decisão compostas

Aninhamento de estruturas de decisão

Aula 1

[SIS]ANO1C1B2S12A1

Exposição

Objetivos da Aula

- Conhecer **estruturas de decisão compostas** por meio do fluxo de execução de programas;
- Compreender exemplos de **aninhamentos em estruturas de decisão**.

Competências da Unidade (Técnicas e Socioemocionais)

- Desenvolver **sistemas computacionais** utilizando ambientes de desenvolvimento;
- Migrar sistemas, implementando **rotinas e estruturas de dados mais eficazes**;
- Trabalhar a **curiosidade e a resiliência** em resolução de problemas computacionais.

Recursos Didáticos

- Recursos audiovisuais para exibição de vídeos e imagens;
- Caderno, canetas e lápis.

Duração da Aula

50 minutos

Exposição

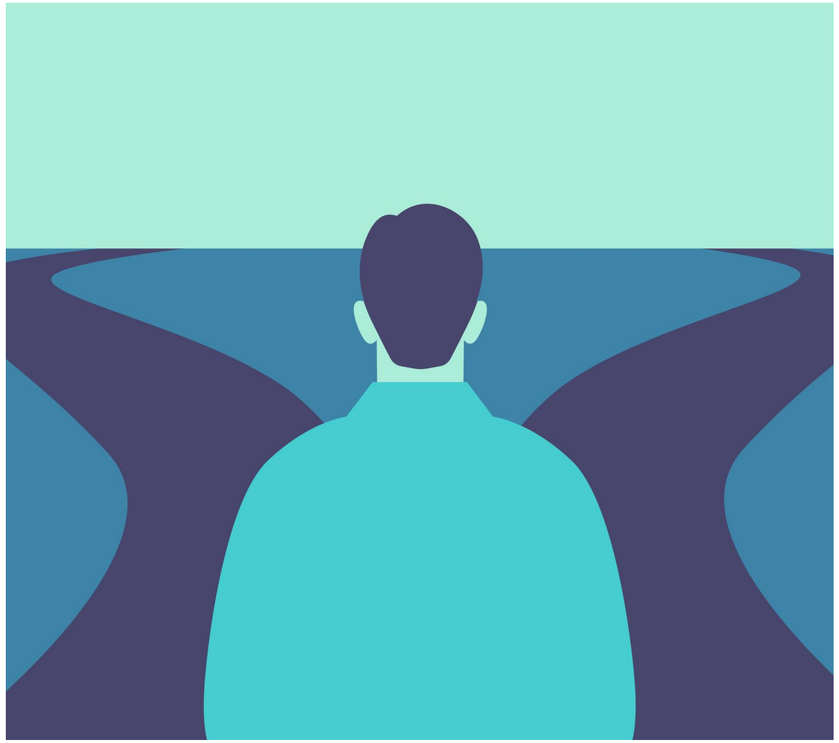
Desenvolvimento da aula

Olá! Hoje, exploraremos um conceito importante em programação chamado **aninhamento** de estruturas de decisão. Entenderemos como ele funciona e como ele pode nos ajudar a escrever programas mais complexos e poderosos.

- Conceito e utilidade do **aninhamento de estruturas de decisão**;
- Análise do **fluxo de execução de um programa** com estruturas de decisão aninhadas;
- Habilidade de escrever **programas com estruturas de decisão aninhadas.**

Exposição

Introdução



© Getty Images

As **estruturas de decisão compostas** nos permitem tomar decisões com base em várias condições.

Nos casos em que precisamos considerar múltiplos níveis de condições, é útil aninhar uma estrutura de decisão dentro de outra.

No **aninhamento de estruturas de decisão**, uma estrutura de decisão (como um bloco *if*, *elif* ou *else*) é colocada dentro de outra estrutura de decisão.

Exposição

Exemplo

```
if condition1:
    if condition2:
        action1
    else:
        action2
else:
    action3
```

Neste exemplo, **condition2** só será verificada se **condition1** for verdadeira.

Se ambas as condições forem verdadeiras, **action1** será executada.

Se **condition1** for verdadeira, mas **condition2** for falsa, **action2** será executada.

Se **condition1** for falsa, **action3** será executada, independentemente do valor de **condition2**.

Exposição

Fluxo de execução

Assim como nas **estruturas de decisão compostas simples**, o fluxo de execução em **estruturas de decisão aninhadas** prossegue de cima para baixo e de dentro para fora.

Quando *Python* encontra uma estrutura de decisão aninhada, ele primeiro avalia a condição da estrutura de decisão externa. Se essa condição for verdadeira, ele **se move para a estrutura de decisão interna e avalia a condição dessa estrutura**.

Importante

Esse fluxo permite que o programa tome decisões complexas baseadas em múltiplos níveis de condições.

Exposição

Exemplos de aninhamento

```
num = int(input("Digite um número: "))
if num > 0:
    if num % 2 == 0:
        print("O número é positivo e par.")
    else:
        print("O número é positivo e ímpar.")
elif num < 0:
    if num % 2 == 0:
        print("O número é negativo e par.")
    else:
        print("O número é negativo e ímpar.")
else:
    print("O número é zero.")
```

Nesse exemplo, o programa pede ao usuário para **inserir um número**.

Queremos imprimir mensagens diferentes, dependendo do número ser:

- positivo e par;
- positivo e ímpar;
- negativo e par;
- negativo e ímpar;
- zero.

Exposição

Exemplos de aninhamento

© Getty Images

Vamos considerar um **exemplo de uma loja que vende alguns produtos**. Queremos fornecer **descontos** com base no tipo de produto e no número de unidades compradas.

- Para produtos de tecnologia, damos um **desconto de 10%** se o cliente comprar **mais de 2 itens;**
- Para produtos de vestuário, damos um **desconto de 20%** se o cliente comprar **mais de 3 itens.**

Exposição

Exemplos de aninhamento

```
tipo_produto = input("Insira o tipo de produto (tecnologia/vestuario): ")
quantidade = int(input("Insira a quantidade de produtos: "))
if tipo_produto == "tecnologia":
    if quantidade > 2:
        print("Você ganhou um desconto de 10%!")
    else:
        print("Infelizmente, você não atingiu a quantidade necessária para um desconto.")
elif tipo_produto == "vestuario":
    if quantidade > 3:
        print("Você ganhou um desconto de 20%!")
    else:
        print("Infelizmente, você não atingiu a quantidade necessária para um desconto.")
else:
    print("Tipo de produto não reconhecido.")
```

Análise de código

A função de **análise de código é muito importante nas empresas** para garantir que o programa desenvolvido esteja dentro dos padrões solicitados. E, caso seja necessário efetuar uma correção, facilita a compreensão da função do algoritmo.

Analise o código abaixo e encontre o que está fora do padrão, impedindo que o programa funcione adequadamente:

```
numeros = [1, 2, 3, 4, 'cinco']

# Filtrando apenas os números da lista
numeros_filtrados = [num for num in numeros if isinstance(num, (int, float))]

# Calculando a soma e a quantidade de números
soma = sum(numeros_filtrados)
quantidade = len(numeros_filtrados)

# Calculando a média
if quantidade > 0:
    media = soma / quantidade
else:
    media = None

print(media)
```

Produção de texto reflexivo

Depois de analisar o código apresentado e encontrar o que estava fora do padrão, impedindo o programa de funcionar:

1. **Elabore um texto** justificando porque determinado ponto está fora do padrão.

2. **Reescreva o código** de modo que o programa funcione corretamente.

Siga os passos do **próximo *slide* para realizar a atividade.**

Registro

Vamos fazer uma **atividade**

Produção de texto reflexivo

O texto deve ter entre 250 e 500 caracteres com espaços (o equivalente a 5 ou 10 linhas).

20 minutos

Pense profundamente sobre o tema e as questões apresentadas. Permita-se ter um tempo para refletir antes de começar a escrever.

Seu texto deve expressar **seu próprio pensamento e perspectiva**. Evite repetir opiniões e ideias de outras pessoas.

Estruture seu texto de **maneira clara e lógica**. Comece com uma introdução ao tema, seguida de suas reflexões, e conclua com uma ideia final.

Use **linguagem clara e compreensível**. Antes de entregar, revise seu texto para garantir a correção gramatical, a clareza e a coerência das ideias.

Registro

Vamos fazer um **quiz**

O termo *else* no fluxo do código pode ser interpretado de qual forma?

- Consequência
- Finalização
- Início
- Repetição

Registro

Vamos fazer um **quiz**

O termo *else* no fluxo do código pode ser interpretado de qual forma?

- ✓ **Consequência**
- **Finalização**
- **Início**
- **Repetição**

RESPOSTA CORRETA!

A expressão *else* é compreendida pelo código como uma consequência no processo de tomada de decisão.

Registro

Vamos fazer um **quiz**

Lembrando sobre os conceitos de operadores lógicos que estudamos na última unidade, qual a importância do operador >=?

- Definir limites comparativos
- Abertura do código
- Finalização do código
- Limpeza de dados

Registro

Vamos fazer um **quiz**

Lembrando sobre os conceitos de operadores lógicos que estudamos na última unidade, qual a importância do operador >=?

- ✓ **Definir limites comparativos**
- **Abertura do código**
- **Finalização do código**
- **Limpeza de dados**

RESPOSTA CORRETA!

O operador >= é utilizado para definir limites comparativos para obter valores maiores ou iguais ao que estamos comparando.

© Getty Images

O que nós **aprendemos hoje?**

Hoje desenvolvemos:

1. O conhecimento mais detalhado sobre **a utilização de estruturas de decisão aninhadas;**

2. A identificação do processo de **tomada de decisões a partir de fluxos;**

3. A prática da **refatoração** da estrutura de um programa em **Python**, utilizando os conceitos compreendidos.

Saiba mais

Que tal aprender um pouco mais sobre o universo das **linguagens de programação**?

Para isso, assista ao seguinte vídeo:

MANUAL DO MUNDO. *Como funcionam as linguagens de programação #SagadosComputadores* Ep. 8, [s.d.]. Disponível em: https://www.youtube.com/watch?v=22nd99SLgNA. Acesso em: 10 mar. 2024.

Referências da aula

MANUAL DO MUNDO. *Como funcionam as linguagens de programação* #SagadosComputadores Ep. 8 , [s.d.]. Disponível em: https://www.youtube.com/watch?v=22nd99SLgNA Acesso em: 10 mar. 2024.

REIS, F. Estrutura de Decisão Condicional Composta if ... else em Python. *Bóson treinamentos em Ciência e Tecnologia*, 2022. Disponível em: https://www.bosontreinamentos.com.br/programacao-em-python/10-python-estrutura-de-decisao-condicional-composta-se-entao-senao/#:~:text=Em%20Python%2C%20outra%20estrutura%20de,a%20condi%C3%A7%C3%A3o%20n%C3%A3o%20for%20atendida Acesso em: 10 mar. 2024.

Educação Profissional Paulista

Técnico em
Desenvolvimento de Sistemas

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO