Educação Profissional Paulista

Técnico em Desenvolvimento de Sistemas

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO

Estruturas de repetição

Conceituação e laço de repetição: *while*

Aula 3

Código da aula: [SIS]ANO1C1B1S5A3

Exposição

Objetivo da aula

Conhecer maneiras performáticas de otimizar a utilização da estrutura de repetição *while*.

Competências da Unidade (técnicas e socioemocionais)

- Desenvolver sistemas computacionais utilizando ambiente de desenvolvimento;
- Resolver problemas computacionais com estratégias ágeis.

Recursos didáticos

- Recurso audiovisual para a exibição de vídeos e imagens;
- Caderno para anotações.

Duração da aula

50 minutos.

Exposição

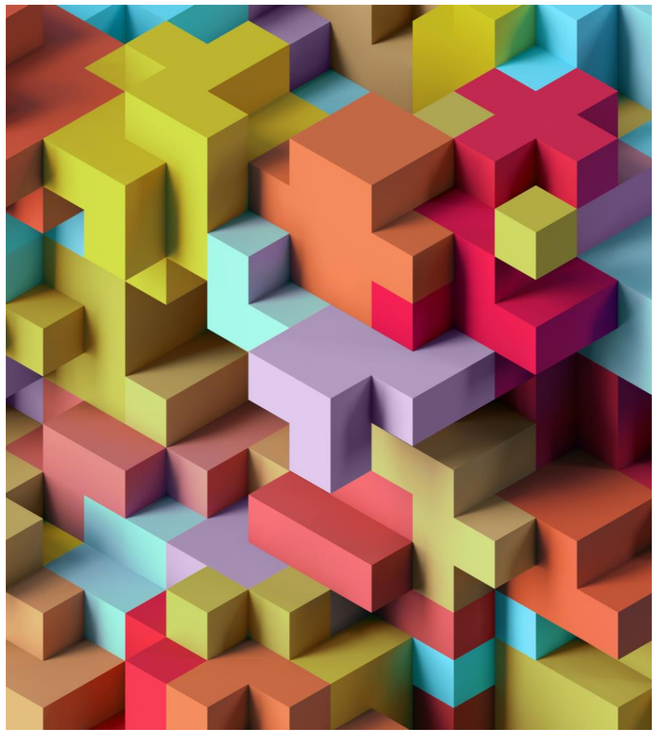
Eficiência e otimização do laço *while*

A otimização do laço while envolve técnicas que aprimoram o desempenho dos programas e economizam recursos, especialmente em loops que processam grandes volumes de dados. Dentre essas técnicas, analisaremos as que envolvem:

- Aninhamento e escopo de variáveis.
- Redução do tempo de execução e uso de memória.
- Depuração para laços while.

Exposição

Aninhamento e escopo de variáveis



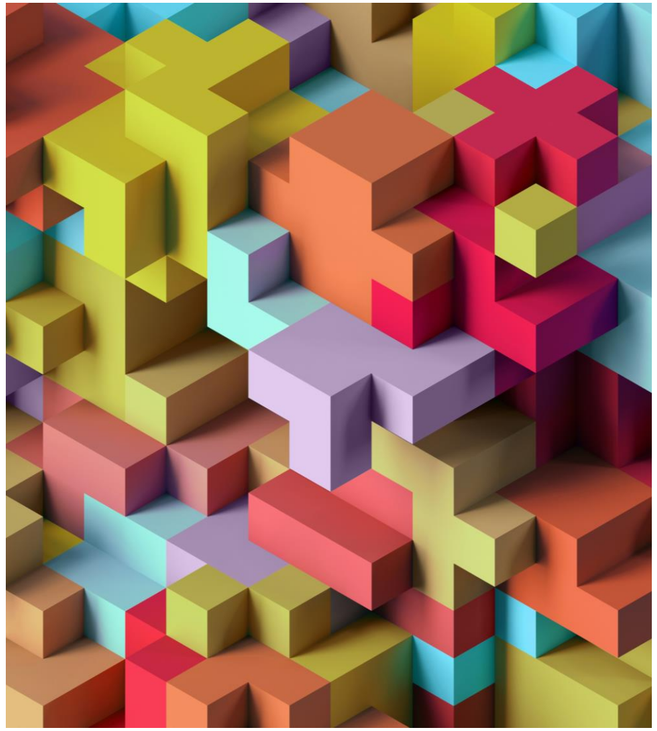
© Getty Images

O aninhamento ocorre quando uma estrutura de controle é colocada dentro de outra, como um laço *while* dentro de outro laço *while* ou dentro de uma instrução *if*. Isso é útil quando se lida com situações que exigem várias camadas de decisão ou repetição.

O escopo de uma **variável** refere-se à parte do programa em que a variável é acessível. Variáveis definidas dentro de um laço têm escopo local; elas só podem ser acessadas dentro desse laço. Por outro lado, variáveis definidas fora de todos os laços têm escopo global, podendo ser acessadas de qualquer lugar do programa.

Exposição

Aninhamento e escopo de variáveis



© Getty Images

Exemplo de escopo em Python:

```
x = "global"

def func():
    y = "local"
    print(x)  # Pode acessar x
    print(y)  # Pode acessar y apenas dentro de func()

func()
print(x)      # Pode acessar x
# print(y)    # Erro! y não está acessível aqui, pois seu
  escopo é local a func()
```

Exposição

Técnicas de depuração para laços *while*

Depurar laços *while* pode ser desafiador, especialmente se eles não estão funcionando como esperado. Aqui estão algumas técnicas:

- ***Print statements*:** coloque declarações de impressão dentro do laço para mostrar o estado atual das variáveis em cada iteração. Isso pode ajudar a entender como o estado do programa muda ao longo do tempo.
- **Condições de saída claras:** certifique-se de que sua condição de saída é alcançável e faça com que as alterações de estado que caminham para essa condição sejam claras e progressivas.
- **Evite alterações complexas de estado:** se a lógica dentro do seu laço é complicada, tente simplificá-la. Se várias variáveis estão envolvidas na decisão de continuação do laço, considere se elas podem ser reduzidas ou calculadas de forma mais simples.

Tome nota

Depuração (ou *debugging*, em inglês) é o processo de identificar e corrigir erros ou bugs em um programa de software. É uma parte crucial do desenvolvimento de software, pois permite que os programadores encontrem e solucionem problemas que podem impedir o programa de funcionar corretamente ou de maneira eficiente.

Exposição

Técnicas de depuração para laços *while*

© Getty Images

Depurar laços *while* pode ser desafiador, especialmente se eles não estão funcionando como esperado. Aqui estão algumas técnicas:

- **Use um depurador:** ferramentas de depuração permitem que você pause a execução do programa, inspecione variáveis e avance passo a passo através do código.
- **Refatoração:** se você encontrar um laço particularmente complexo ou propenso a erros, pode ser um sinal de que o laço precisa ser refatorado. Talvez você possa dividir o laço em funções menores ou reescrever a lógica de uma forma que faça mais sentido.
- **Testes unitários:** escreva testes unitários que verifiquem o comportamento do laço com diferentes entradas e condições para garantir que ele funcione como esperado em vários cenários.

Exposição

Avaliação de performance de laços *while*

© Getty Images

A avaliação de performance de laços *while* envolve medir o tempo e o uso de recursos que o laço consome durante a execução.

A eficiência de um laço *while* pode ser crucial, especialmente se ele for destinado a processar uma grande quantidade de dados ou for executado frequentemente.

Avaliação de performance de laços *while*

Exemplo prático:
Suponhamos que você tenha um laço *while* que processa elementos de uma lista e remove itens que atendem a um certo critério:

```
import time

# Lista com números de 0 a 999999
lista = list(range(1000000))

start_time = time.time()  # Tempo inicial

# Laço while para remover números ímpares
i = 0
while i < len(lista):
    if lista[i] % 2 != 0:
        lista.pop(i)
    else:
        i += 1

end_time = time.time()  # Tempo final

print(f"Tempo de execução: {end_time - start_time} segundos.")
```

Exposição

Refatoração de *loops while* existentes

Refatoração é o processo de alterar a estrutura do código para melhorar sua leitura, manutenção ou performance sem alterar seu comportamento externo. Laços *while* podem muitas vezes ser refatorados para aumentar a eficiência.

Exemplo de refatoração:
Vamos refatorar o código anterior para melhorar a performance:

```
# ... (código anterior) ...

start_time = time.time()  # Tempo inicial

# Laço while refatorado para usar list comprehension
lista = [x for x in lista if x % 2 == 0]  # Mantém apenas números pares

end_time = time.time()  # Tempo final

# ... (código anterior) ...
```

© Getty Images

Exposição

Vídeo

Que tal conhecermos mais um exemplo da aplicação das estruturas de repetição em Python?

CURSOS KANE CHAN. Estrutura de Repetição While em Python. Disponível em: https://www.youtube.com/watch?v=iFYWrDMfVNo Acesso em: 26 jan. 2024.

O que nós **aprendemos hoje?**

Hoje desenvolvemos:

1. O funcionamento da estrutura de aninhamento e escopo de variáveis.

2. O estudo das técnicas de depuração para laços *while* e seu funcionamento.

3. A estrutura de avaliação de performance para laços *while*.

Saiba mais

Durante esta aula, falamos sobre técnicas aplicadas no conceito de depuração de código. Que tal aprendermos mais a fundo o que é esse conceito?

“O que é debugar e como aprender a programar mais rápido”, do canal MICAEL MOTA. Disponível em: https://www.youtube.com/watch?v=5nwwwDu4ZzE. Acesso em: 26 jan. 2024.

Referências da aula

CURSOS KANE CHAN. **Estrutura de Repetição While em Python**. Disponível em: https://www.youtube.com/watch?v=iFYWrDMfVNo. Acesso em: 26 jan. 2023.

LOPES, E. Loops e estruturas de repetição no Python. **Python Academy**, 8 jul. 2021. Disponível em: https://pythonacademy.com.br/blog/estruturas-de-repeticao. Acesso em: 26 jan. 2024.

Identidade visual: imagens © Getty Images

Educação Profissional Paulista

Técnico em Desenvolvimento de Sistemas

Secretaria da Educação

SÃO PAULO
GOVERNO DO ESTADO