
IN
INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

IN AULA 04 | LÓGICA DE PROGRAMAÇÃO

LP – Estrutura de Repetição: WHILE

01 O loop While

02 Um loop infinito

03 As instruções break e continue

IN AULA 04 | LÓGICA DE PROGRAMAÇÃO

LP – Estrutura de Repetição: WHILE

Em Python, existem dois tipos de *loops*: o **FOR** e o **WHILE**. Nesta aula, entenderemos o funcionamento do laço **WHILE**.

O while segue uma lógica bem simples:

enquanto há algo para fazer
    faça isso

Observe que, se não houver nada a fazer, nada acontecerá.
O while possui ainda algumas semelhanças com a instrução if, tudo bem. Na verdade, a diferença é apenas uma: você usa a palavra while em vez da palavra if.

Lógica do if:

se há algo para fazer
    faça isso

LP – Estrutura de Repetição: WHILE

A diferença semântica entre eles carrega um detalhe bem específico:

- Na instrução if, quando a condição é atendida, se executa suas instruções apenas uma vez.
- Na instrução while, enquanto a condição é avaliada como True, as instruções são repetidas.

Em geral, o *loop* while pode ser representado da seguinte forma:

```
while condição:
    instrução
```

Uma instrução ou um conjunto de instruções executadas dentro do loop é chamado de **corpo do *loop***. Se a condição for False quando for testada pela primeira vez, o corpo não será executado nenhuma vez.

obs: Todas as regras relacionadas à indentação também são aplicáveis ao while.

LP – Estrutura de Repetição: WHILE

```
# Declaramos uma variável e atribuímos a ela o valor 0 (int)
contador = 0

# Implementamos o loop while utilizando a
# variável contador para controlar a condição
# de progresso do nosso loop
while contador < 3:
    print(contador)
    contador = contador + 1
```

obs: O corpo deve ser capaz de alterar o valor da condição, porque se a condição for True no início, o corpo poderá ser executado continuamente até o infinito.

EXEMPLO

No primeiro momento, podemos observar uma condição que se mantém verdadeira continuamente enquanto a variável contador é 0.

Graças ao poder de repetição do *loop*, ele pode automaticamente realizar as instruções no seu corpo fazendo com que a variável contador seja imprimida e somada a 1.

Quando a variável contador chegar ao valor 3, a condição no *loop* se tornará falsa e então ele não mais será repetido.

IN AULA 04 | LÓGICA DE PROGRAMAÇÃO

LP – Estrutura de Repetição: WHILE

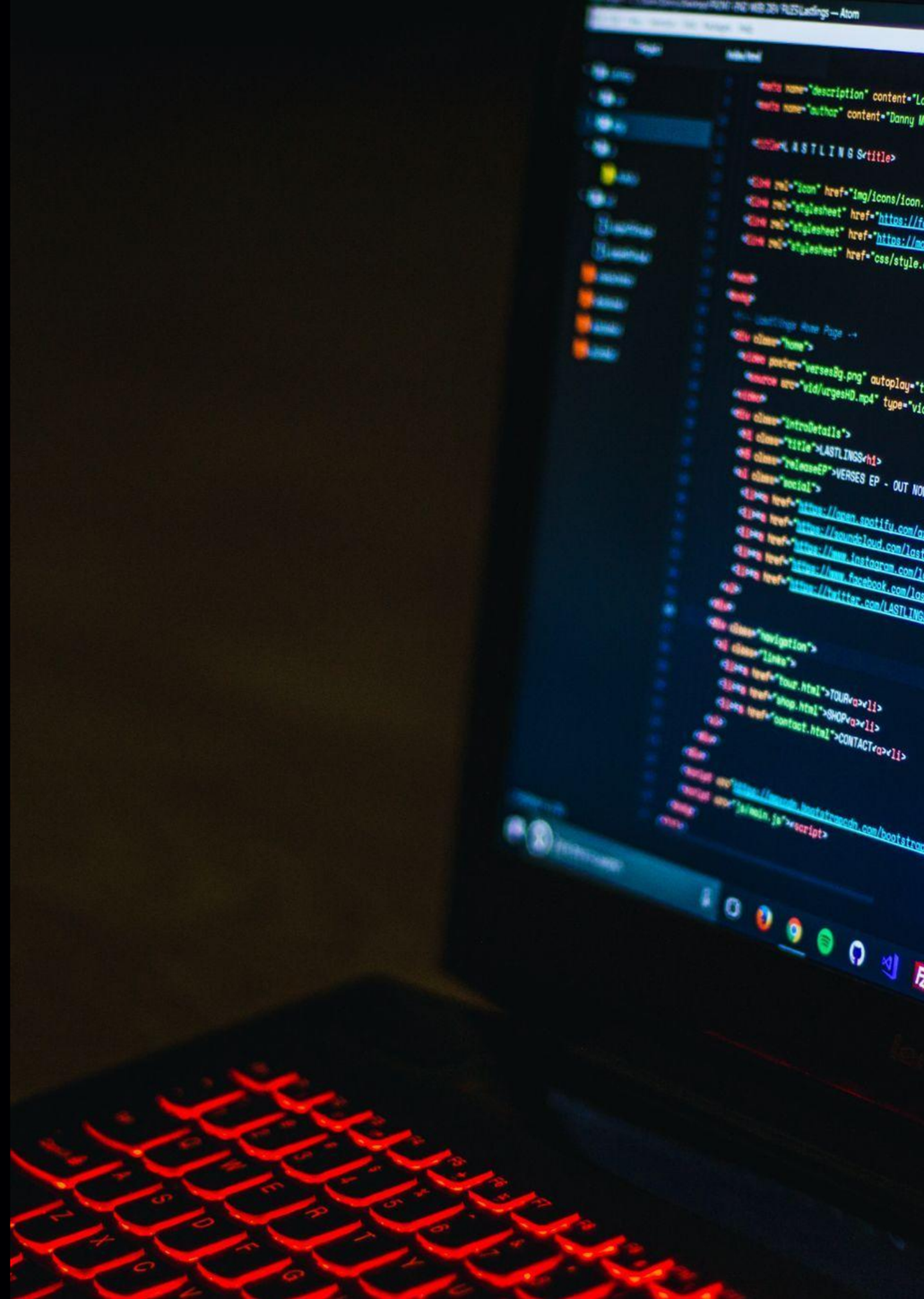
Um *loop* infinito

Um *loop* infinito é uma sequência de instruções em um programa que se repete indefinidamente.

```
while True:
    print("Estou preso dentro de um loop.")
```

Este loop exibirá infinitamente "Estou preso dentro de um loop." na tela.

Mas. . . Como sair então desse *loop* infinito?



LP – Estrutura de Repetição: WHILE

Até agora, tratamos o corpo do loop como uma sequência indivisível e inseparável de instruções que são executadas completamente a cada turno do loop. No entanto, como desenvolvedor, você pode se deparar com as seguintes situações:

- parece que é desnecessário continuar o *loop* como um todo; você deve se abster de executar o corpo do *loop* e ir além;

- parece que você precisa iniciar a próxima curva do *loop* sem concluir a execução da curva atual.

LP – Estrutura de Repetição: WHILE

As instruções break e continue

O Python possui duas instruções especiais para a resolução dos problemas abordados no slide anterior, são elas:

break : O break só pode ocorrer sintaticamente aninhado em um laço de repetição, mas não aninhado em uma função ou definição de classe dentro desse laço. Ao ser ativado ele para completamente a execução do loop seguindo para o próximo comando fora do loop ou encerrando caso não tenha mais nada. Ele termina o laço de fechamento mais próximo, pulando a cláusula opcional else se o laço tiver uma. Se um laço for é encerrado por break, o alvo de controle do laço mantém seu valor atual.

continue : O continue se comporta como se o programa tivesse chegado ao fim do corpo; Todo o bloco de código abaixo dele é ignorado e o próximo turno é iniciado.

LP – Estrutura de Repetição: WHILE

break :

```
# Criando a variável contador
contador = 0
# Iniciando o loop com a condição True
while True:
    # Colocando uma condição para o uso do break
    if contador == 10:
        break
    # Somando 1 ao total da variável contador
    else:
        contador = contador + 1
```

EXPLICAÇÃO

No início do código nós podemos observar a criação da variável contador para dar utilidade ao nosso exemplo.
O loop começa com a palavra chave True tornando o loop infinito. Logo abaixo podemos observar a condicional if que é ativada no momento em que o contador possui o valor 10.
Ao ser ativado o break imediatamente para o programa encerrando o loop infinito.

LP – Estrutura de Repetição: WHILE

continue :

```
# Criando a variável contador
contador = 0
# Iniciando o loop com a condição True
while True:
    # Somando 1 ao total da variável contador
    contador = contador + 1
    # Colocando uma condição para o uso do break
    if contador == 10:
        break
    # Criando uma condição para o uso do continue
    elif contador == 3:
        continue
    print(contador)
```

EXPLICAÇÃO

Aqui alteramos o programa anterior para a utilização do continue.

Ao rodar o programa você vai perceber que a saída é um pouco diferente: 1 2 4 5 6 7 8 9

Podemos notar que o número 3 não está presente na saída. Isso acontece porque ao chegar no valor de número 3, o contador atendeu a condição visível no elif.

Ao chegar neste ponto, o continue foi ativado ignorando o restante do código, o que impediu que o número 3 fosse imprimido na tela.

LP – Estrutura de Repetição: WHILE

A cláusula else

Ambos os loops, while e for, possuem um recurso interessante e raramente utilizado. Os loops também podem ter o ramo else. Ele se encontra sempre depois do loop e sempre é executado ao menos uma vez independente de o loop ter entrado no seu corpo ou não.

```
numero = 1
while numero < 3:
    print(numero)
    numero += 1
else:
    print("else:", numero)
```

```
for numero in range(5):
    print(numero)
else:
    print("else:", numero)
```

LP – Estrutura de Repetição: WHILE

1. Faça um programa que peça uma nota, entre zero e dez. Mostre uma mensagem caso o valor seja inválido e continue pedindo até que o usuário informe um valor válido.
2. Faça um programa que leia um nome de usuário e a sua senha e não aceite a senha igual ao nome do usuário, mostrando uma mensagem de erro e voltando a pedir as informações.
3. Faça um programa que leia e valide as seguintes informações:
- Nome: maior que 3 caracteres;
- Idade: entre 0 e 150;
- Salário: maior que zero;
- Sexo: 'f' ou 'm';
- Estado Civil: 's', 'c', 'v', 'd';

4. Supondo que a população de um país A seja da ordem de 80000 habitantes com uma taxa anual de crescimento de 3% e que a população de B seja 200000 habitantes com uma taxa de crescimento de 1.5%. Faça um programa que calcule e escreva o número de anos necessários para que a população do país A ultrapasse ou iguale a população do país B, mantidas as taxas de crescimento.
5. Altere o programa anterior permitindo ao usuário informar as populações e as taxas de crescimento iniciais. Valide a entrada e permita repetir a operação.
6. Faça um programa que imprima na tela os números de 1 a 20, um abaixo do outro. Depois modifique o programa para que ele mostre os números um ao lado do outro.
7. Faça um programa que leia 5 números e informe o maior número.
8. Faça um programa que leia 5 números e informe a soma e a média dos números.
9. Faça um programa que imprima na tela apenas os números ímpares entre 1 e 50.
10. Faça um programa que receba dois números inteiros e gere os números inteiros que estão no intervalo compreendido por eles.