



PYTHON  
ACADEMY

# ESTRUTURAS DE REPETIÇÃO USANDO LOOPS WHILE NO PYTHON

Nesse ebook você vai aprender a utilizar loops while do Python para criar Estruturas de Repetição.

[PYTHONACADEMY.COM.BR](https://pythonacademy.com.br)

Este ebook foi gerado por



# Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs

Deixe que nossa IA faça o trabalho pesado

 Syntax Highlight

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

**TESTE AGORA** 

 PRIMEIRO CAPÍTULO 100% GRÁTIS

Seja muito bem-vindo Pythonista!

Nesse artigo, você vai aprender a criar estruturas de repetição utilizando o `while` do Python.

Estruturas de repetição - como o próprio nome já diz - são trechos de código onde você precisa aplicar determinados comandos repetidas vezes.

Eles são utilizados constantemente em códigos Python, você verá no seu dia a dia de Pythonista! 😊

Está preparado?! Então vamos nessa! 🚀

## Estruturas de repetição

**Loops** ou **estruturas de repetição** são blocos básicos de qualquer linguagem de programação e são muito importantes!

Cada linguagem de programação possui uma sintaxe específica para criação destas *loops*.

Vamos ver nesse post como podemos fazer loops utilizando o `while` !

É essencial **DOMINAR** essa estrutura de repetição para se tornar um verdadeiro Pythonista!

## Loops utilizando while

O `while` é uma estrutura de repetição utilizada quando queremos que determinado bloco de código seja executado **ENQUANTO** (do inglês *while*) determinada condição for satisfeita.

Em outras palavras:

*“Só saia da estrutura de repetição quando a condição não for mais satisfeita”*

Sua sintaxe básica é:

```
while {condição}:  
    {código}
```

Vamos entender cada pedaço dessa sintaxe:

No `while`, a parte `{condição}` é uma expressão que pode ser reduzida à `True` ou `False`, podendo ser: - A verificação de valor de uma variável; - Determinada estrutura alcançar um tamanho; - O retorno de uma função se igualar a determinado valor;

Já `{código}` vai ser o bloco de código a ser repetido a cada iteração do loop `while` !

Vamos entender melhor com um exemplo:

```
contador = 0  
  
while contador < 5:  
    print(f'Valor do contador é {contador}')    contador += 1
```

O código resultará em:

```
Valor do contador é 1
Valor do contador é 2
Valor do contador é 3
Valor do contador é 4
Valor do contador é 5
```

Ou seja, a variável `contador` está sendo incrementada a cada vez que o `while` executa seu código.

Quando ele alcançar o valor 5, a condição `contador < 5` não será mais satisfeita, finalizando o loop `while` !

## ○ `break` e ○ `continue`

Existem duas palavras reservadas da linguagem que servem para auxiliar no controle do fluxo da estrutura de repetição. São elas: `break` e o `continue` !

Utilizamos o `break` para parar a execução de um loop. Geralmente utilizamos uma estrutura condicional com `if` para então usar a cláusula `break` !

Já o `continue` é utilizado para pular todo código que estiver após esta cláusula, levando em frente na próxima iteração do *loop*!

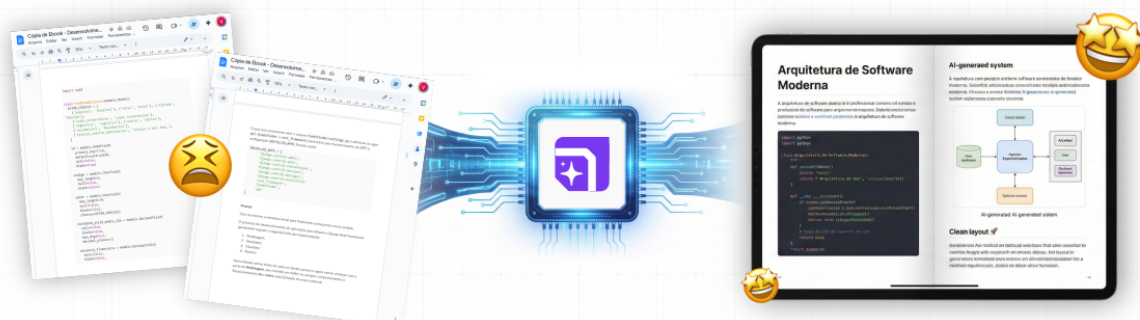


*Estou construindo o **DevBook**, uma plataforma que usa IA para criar ebooks técnicos — com código formatado e exportação em PDF. Depois de ler, dá uma passada lá!*



## Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs

Deixe que nossa IA faça o trabalho pesado

Syntax Highlight

Adicione Banners Promocionais

Edite em Markdown em Tempo Real

Infográficos feitos por IA

TESTE AGORA! PRIMEIRO CAPÍTULO 100% GRÁTIS

## Loops utilizando `while` e `else`

Podemos ainda adicionar a cláusula `else` em loops `while` !

Sim! Este é um fato que muitos Pythonistas desconhecem.

O `else` nos possibilita executar um bloco de código após a condição ter sido satisfeita.

Porém, o `else` não é executado quando o `while` encontra uma cláusula `break` !

Vamos entender melhor no exemplo:

```
import random

numero_magico = random.randint(0,100)
tentativas = 0

while tentativas < 3:
    numero = input('Adivinhe o número mágico (0 a 100): ')

    if int(numero) == numero_magico:
        print('Corre pra Loteria! Hoje é seu dia de sorte *.*')
        break
    tentativas += 1
else:
    print('Teeeente outra veeeez xD')
```

Nesse exemplo, perguntamos um número ao usuário e ele deve acertar o número randômico gerado pelo programa em menos de 3 tentativas.

Se ele acertar, o texto **“Corre pra Loteria! Hoje é seu dia de sorte \*.\*”** deve ser mostrado!

Caso contrário, se ele não acertar em 3 tentativas, o seguinte texto deverá ser mostrado: **“Teeeente outra veeeez xD”**

Perceba que o `else` não deve ser executado caso o código passe pela cláusula `break` !

## Conclusão

Agora que você já sabe o funcionamento das estruturas de repetição utilizando `for` teste-as e verifique seus usos em diferentes casos, garanto que você irá aprender bastante!

Qualquer dúvida fique à vontade para utilizar o box de comentários abaixo!

Nos vemos na próxima! 😊



Não se esqueça de conferir!



DevBook

# Crie Ebooks técnicos em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs



 Syntax Highlight

 Infográficos feitos por IA

 Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado

 Edite em Markdown em Tempo Real

**TESTE AGORA** 

 PRIMEIRO CAPÍTULO 100% GRÁTIS