

Repetições em Python

Repetições representam a base de vários programas. São utilizadas para executar a mesma parte de um programa várias vezes, normalmente dependendo de uma condição. Imagine imprimir 100 números - sem repetições, precisaríamos de pelo menos 200 linhas de código!

A estrutura de repetição aparece para nos auxiliar a resolver esse tipo de problema de forma elegante e eficiente.



A Estrutura While

Formato Básico

```
while <condição>:  
    bloco
```

Funcionamento

Repete um bloco enquanto a condição for verdadeira

Controle

A condição é avaliada antes de cada execução do bloco

```
x = 1  
while x <= 3:  
    print(x)  
    x = x + 1
```

Exemplo Prático: Imprimindo de 1 a 3

Sem While (Repetitivo)

```
print(1)  
print(2)  
print(3)
```

Imagine fazer isso para 100 números! --

Com While (Elegante)

```
x = 1  
while x <= 3:  
    print(x)  
    x = x + 1
```

Facilmente adaptável para qualquer quantidade

Fluxo de Execução do While

01

Inicialização

$x = 1$ (primeira execução)

02

Verificação da Condição

$x \leq 3$ é verdadeiro?

03

Execução do Bloco

print(x) e $x = x + 1$

04

Retorno

Volta para verificar a condição novamente

05

Término

Quando $x = 4$, condição fica falsa

Exercícios Básicos

Exercício 5.1

Modifique o programa para exibir os números de 1 a 100

Exercício 5.2

Modifique o programa para exibir os números de 50 a 100

Exercício 5.3

Faça um programa para a contagem regressiva de um foguete: 10, 9, 8, ..., 1, 0, Fogo!

Contadores

Um contador é uma variável utilizada para contar o número de ocorrências de um determinado evento. No contexto de repetições, contamos o número de execuções do while.

```
fim = int(input("Digite o último número a imprimir:"))  
x = 1  
while x <= fim:  
    print(x)  
    x = x + 1
```

A variável **x** é nosso contador, incrementando de 1 em 1 a cada repetição. Quando digitamos 0, nada acontece pois a condição já é falsa desde o início.

Números Pares com Contadores

Solução com IF

```
fim = int(input("Digite o último número:"))  
x = 0  
while x <= fim:  
    if x % 2 == 0:  
        print(x)  
    x = x + 1
```

Solução Otimizada

```
fim = int(input("Digite o último número:"))  
x = 0  
while x <= fim:  
    print(x)  
    x = x + 2
```

Ambas as soluções são corretas! A segunda é mais eficiente pois incrementa diretamente de 2 em 2.

Tabuada Simples

Vamos criar um programa que imprime a tabuada de adição de um número digitado pelo usuário, de 1 a 10:

```
n = int(input("Tabuada de:"))  
x = 1  
while x <= 10:  
    print(n + x)  
    x = x + 1
```

Flexibilidade

O usuário escolhe qual número usar na tabuada

Padrão Fixo

Sempre de 1 a 10, mas pode ser modificado

Exercícios de Contadores

1

Exercício 5.4

Imprimir apenas números ímpares de 1 até um número digitado

2

Exercício 5.5

Escrever os 10 primeiros múltiplos de 3

3

Exercício 5.6

Alterar para formato de tabuada: $2 \times 1 = 2$, $2 \times 2 = 4$...

4

Exercício 5.7

Usuário digita início e fim da tabuada



Exercícios Avançados de Multiplicação

→ Exercício 5.8 - Multiplicação por Soma

Calcular multiplicação usando apenas soma e subtração. Lembre-se: $4 \times 5 = 5 + 5 + 5 + 5$


→ Exercício 5.9 - Divisão por Subtração

Calcular divisão inteira usando apenas soma e subtração. Exemplo: $20 \div 4 = 5$ (subtraímos 4 cinco vezes de 20)

Correção de Teste com Contadores

Exemplo prático: programa para corrigir teste de múltipla escolha com três questões. Respostas corretas: 1-"b", 2-"a", 3-"d".

```
pontos = 0
questão = 1
while questão <= 3:
    resposta = input("Resposta da questão %d: " % questão)
    if questão == 1 and resposta == "b":
        pontos = pontos + 1
    if questão == 2 and resposta == "a":
        pontos = pontos + 1
    if questão == 3 and resposta == "d":
        pontos = pontos + 1
    questão += 1
print("O aluno fez %d ponto(s)" % pontos)
```

 **Atenção:** O programa diferencia maiúsculas de minúsculas. "A" é diferente de "a"!