# Educação Profissional Paulista

Técnico em

Desenvolvimento

de Sistemas



## Estruturas de repetição Laço de repetição: *FOR*

Aula 4

Código da aula: [SIS]ANO1C1B2S14A4





#### Objetivo da Aula

• Conhecer a implementação e o uso do laço *FOR* em diversas linguagens de programação, evidenciando as semelhanças e as diferenças fundamentais.



#### Competências da Unidade (Técnicas e Socioemocionais)

- Desenvolver sistemas computacionais, utilizando ambiente de desenvolvimento;
- Migrar sistemas, implementando rotinas e estruturas de dados mais eficazes;
- Trabalhar a criatividade e o comprometimento na resolução de problemas computacionais.



#### **Recursos Didáticos**

- Recurso audiovisual para a exibição de vídeos e imagens;
- Folhas sulfite, canetas coloridas, lápis.



#### Duração da Aula

50 minutos

#### Laço FOR em diferentes linguagens de programação

Vamos comparar a implementação e o uso do laço *FOR* em diversas linguagens de programação, evidenciando as semelhanças e as diferenças fundamentais.



Comparação entre linguagens (como Python, Java, C++).



Variações de sintaxe e uso.



Adaptação de algoritmos entre linguagens.





## A etapa de refatoração é crucial. Vamos analisar o que é preciso refatorar no projeto?



ALURA. *Python:* crie a sua primeira aplicação. 07 Refatorando o código. Disponível em:

Acesso em: 28 mar. 2024.



## Comparação entre Python, Java e C++

	Python	Java	C++
Uso	Amplamente usado para desenvolvimento web, análise de dados, inteligência artificial e automação.	Predominante em ambientes corporativos, sistemas Android, aplicativos web e servidores.	Desenvolvimento de sistemas e aplicativos em que o desempenho é crucial, como jogos, aplicativos gráficos e sistemas embarcados.
Características	Linguagem de alto nível, dinâmica, interpretada e de fácil leitura.	Linguagem orientada a objetos, compilada para Bytecode executável em qualquer máquina com Java Virtual Machine (JVM).	Linguagem de médio nível, com suporte para programação orientada a objetos e programação de baixo nível (como manipulação de memória).
Vantagens	Sintaxe clara, vasta biblioteca padrão, ideal para prototipagem rápida e para iniciantes.	Portabilidade, robustez, escalabilidade e um ecossistema rico.	Controle preciso sobre recursos do sistema, eficiência e velocidade.

Elaborado especialmente para o curso.





© Getty Images

#### Variações de sintaxe e uso

- Python é conhecido por sua sintaxe clara e uso de indentação para definir blocos de código, tornando-o muito legível.
- Java utiliza uma sintaxe mais verbosa, com a necessidade de definir classes e métodos explicitamente.
- C++ apresenta uma sintaxe complexa que viabiliza operações de baixo nível, mas pode ser menos intuitiva e mais propensa a erros.

#### Variações de sintaxe e uso

Exemplo da implementação de um algoritmo simples em cada linguagem

#### **Python**

```
def somar(a, b):
    return a + b

# Exemplo de uso de um laço FOR com range
total = 0
for i in range(4): # Isso vai iterar i de 0 a 3
    total = somar(total, i)

print(total)
```



#### Variações de sintaxe e uso

Exemplo da implementação de um algoritmo simples em cada linguagem

#### Java

```
public class Main {
  public static int somar(int a, int b) {
    return a + b;
  public static void main(String[] args) {
    int total = 0;
    for (int i = 0; i < 4; i++) { // Isso vai iterar i de 0 a 3
       total = somar(total, i);
    System.out.println(total);
```

#### Variações de sintaxe e uso

Exemplo da implementação de um algoritmo simples em cada linguagem

#### C++

```
#include <iostream>
using namespace std;
int somar(int a, int b) {
  return a + b;
int main() {
  int total = 0;
  for (int i = 0; i < 4; i++) { // Isso vai iterar i de 0 a 3
    total = somar(total, i);
  cout << total; // Isso imprimirá a soma de 0 + 1 + 2 + 3
  return 0;
```

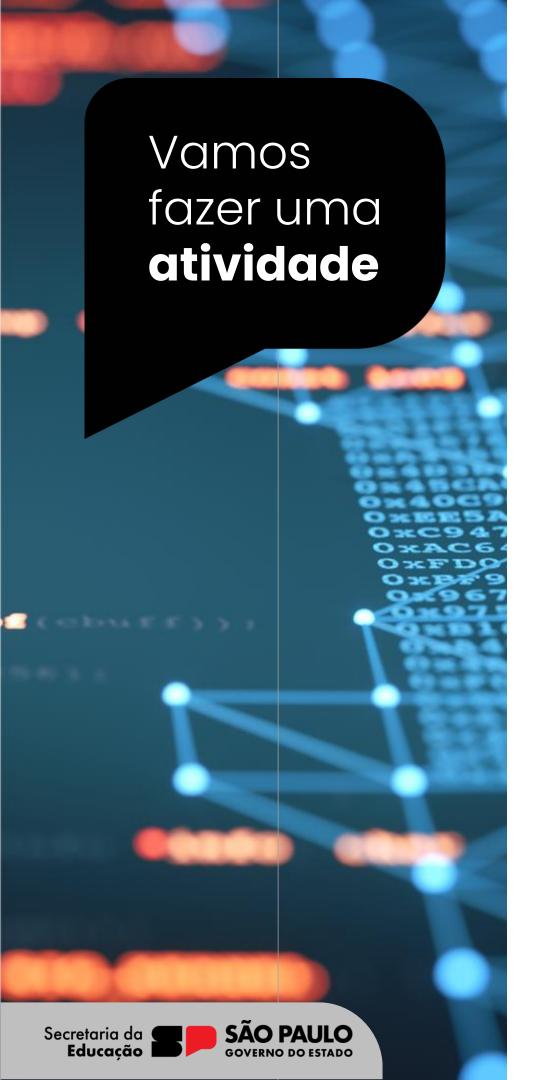


### Adaptação de algoritmos entre linguagens

A adaptação de algoritmos entre linguagens envolve compreender a lógica do algoritmo e então implementá-la, respeitando as peculiaridades de cada linguagem.

Por exemplo, a gestão de memória é um aspecto crucial em C++, enquanto em Python e Java isso é gerenciado automaticamente.





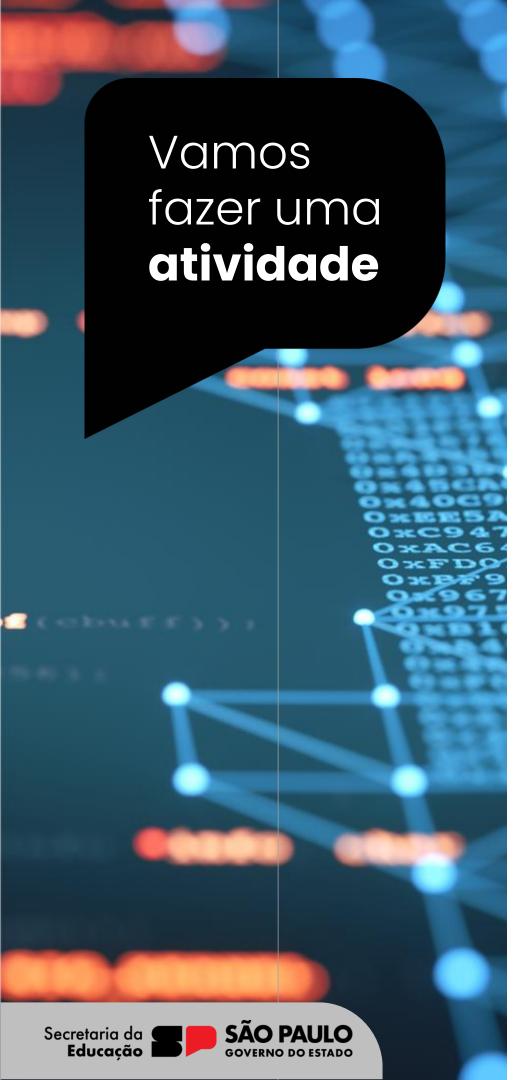
### Entendendo e aplicando o laço *FOR* em Python

#### Objetivo

Compreender e demonstrar como o laço *FOR* pode ser usado para melhorar a eficiência e a clareza do código em Python. Ao final da atividade, os participantes deverão ser capazes de ilustrar como o uso do laço *FOR* pode tornar o código mais eficiente e legível.

#### **Enunciado**

Você faz parte de um grupo de programadores iniciantes em Python. O desafio é reescrever um código que atualmente usa múltiplas instruções repetitivas e substituí-las por um laço *FOR.* O objetivo é tornar o código mais eficiente e fácil de ler.



### Entendendo e aplicando o laço *FOR* em Python

Informações necessárias

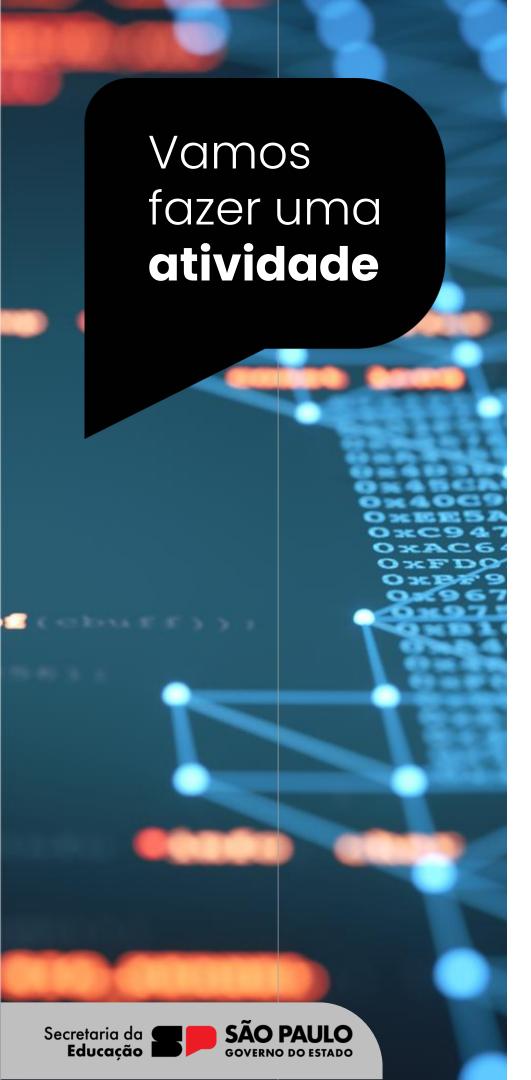
Laço *FOR* em Python: utilizado para iterar sobre uma sequência (que pode ser uma lista, uma tupla, um dicionário, um conjunto ou uma string).

Estrutura básica

for valor in sequencia:

# Bloco de código a ser repetido

- Exemplos de uso: iterar sobre listas, realizar operações repetidas etc.
- Vantagens: reduz a repetição de código e melhora a legibilidade.



### Entendendo e aplicando o laço *FOR* em Python

#### Exemplo de solução

#### Código original

```
print("Número 1")
print("Número 2")
print("Número 3")
print("Número 4")
print("Número 5")
```

#### Código reescrito com FOR

```
for i in range(1, 6):
print(f"Número {i}")
```

Após concluir a tarefa, discutam sobre como o uso do laço *FOR* melhorou o código. Considerem aspectos como a redução do número de linhas de código, a facilidade de manutenção e a clareza da lógica de programação.

#### Vamos fazer uma atividade

## Entendendo e aplicando o laço *FOR* em Python

Compreender e demonstrar como o laço *FOR* pode ser usado para melhorar a eficiência e a clareza do código em Python. Ao final da atividade, os participantes deverão ser capazes de ilustrar como o uso do *FOR* pode tornar o código mais eficiente e legível.



Identificar um código que usa instruções repetitivas (por exemplo, múltiplas instruções de impressão) e reescrevê-lo usando um laço FOR. Criar um esquema simples (pode ser um fluxograma ou pseudocódigo) ilustrando como o laço FOR melhora o desempenho e a legibilidade do código. Após concluir a tarefa, discutam sobre como o uso do laço *FOR* melhorou o código. Considerem aspectos como a redução do número de linhas de código, a facilidade de manutenção e a clareza da lógica de programação. Enviar a atividade, conforme orientação do professor responsável.





#### Hoje desenvolvemos:

A comparação entre linguagens, como Python, Java e C++.

A compreensão das variações de sintaxe e uso das estruturas de repetição.

3 A adaptação de algoritmos entre linguagens de programação.





#### Referências da aula

ALURA. *Python*: crie a sua primeira aplicação. 07 Refatorando o código. Disponível em: <a href="https://cursos.alura.com.br/course/python-crie-sua-primeira-aplicacao/task/146277">https://cursos.alura.com.br/course/python-crie-sua-primeira-aplicacao/task/146277</a>. Acesso em: 28 mar. 2024.

SHARPAX. *Aula 13 – Vetores (Arrays) I Lógica de programação*. Disponível em: <a href="https://www.youtube.com/watch?v=NwllouSVKN4">https://www.youtube.com/watch?v=NwllouSVKN4</a>. Acesso em: 28 mar. 2024.

ZANELALO, J. *Lógica de programação* – Estruturas de repetição. PodProgramar, 25 fev. 2018. Disponível em: <a href="https://podprogramar.com.br/logica-de-programacao-estruturas-de-repeticao/">https://podprogramar.com.br/logica-de-programacao-estruturas-de-repeticao/</a>. Acesso em: 28 mar. 2024.

Identidade visual: imagens © Getty Images.



# Educação Profissional Paulista

Técnico em

Desenvolvimento

de Sistemas

