



Paradigmas de linguagens de programação em python

Professores:

Sebastião Rogerio feat. Kayo Monteiro

Agenda

01

O que são
paradigmas de
programação?

02

Por que aprender
sobre paradigmas
de programação?

03

Quais os principais
paradigmas de
programação?

04

Primeiros passos com
a linguagem python



01

0 que são paradigmas?

O que são paradigmas?

pode ser entendido como um **tipo de estruturação ao qual a linguagem deverá respeitar**. A depender do objetivo proposto, a solução que a linguagem oferecerá obedece a um tipo de paradigma. Portanto, o que vai definir o paradigma utilizado será a tratativa dada ao problema.



Vamos resolver um problema!

Imagine que você tivesse que levar um sofá do primeiro ao terceiro andar do seu apartamento. O problema, já se sabe qual é.

Mas de qual forma (paradigma) resolver essa questão?





E em programação?

O que é um programa?

- Quanto mais **humanamente compreensível** a descrição, mais abstrata será sua implementação numa **linguagem de programação**;
- E como funciona esse processo da programação à execução?
 - Linguagens de alto nível
 - Linguagem de máquina
 - Compiladores / Interpretadores



Linguagens de Programação

Interpretadas X Compiladas



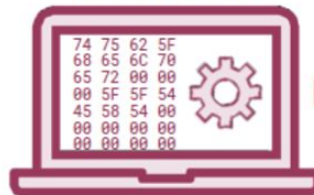
Source code



My computer



Source code



Your computer

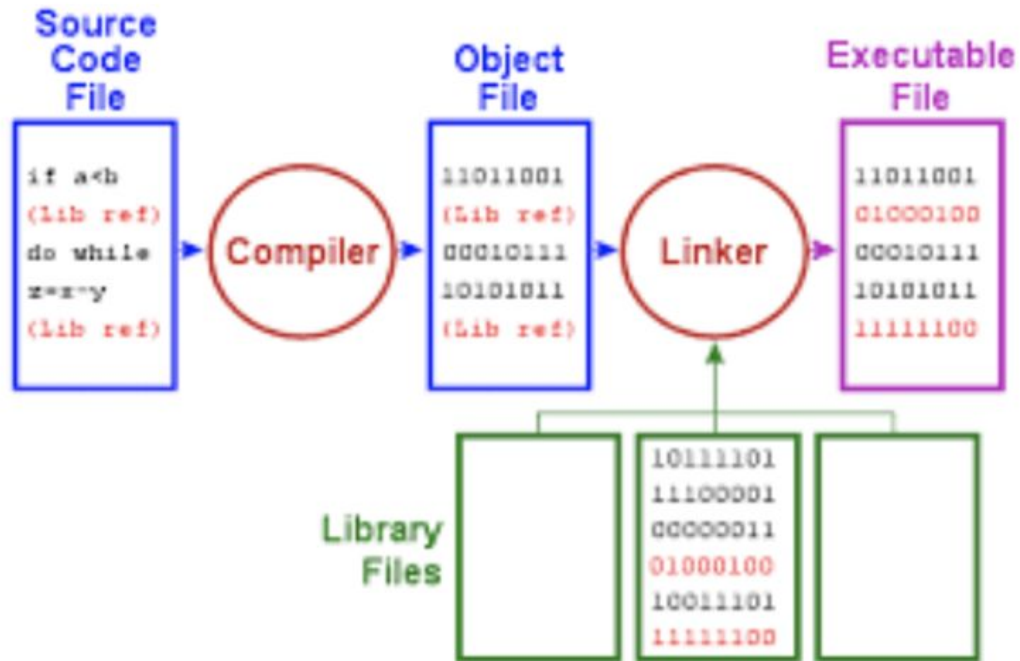


Interpreter

Interpretadas

- Cada comando do código é transformado em comandos de baixo nível, um a um;
- O interpretador é o responsável pela tradução;
- Como a tradução é feita “em tempo real”, comando a comando, normalmente linguagens interpretadas são menos eficientes;
- Linguagens compiladas podem realizar otim levando em conta todo o código;
- Entretanto, linguagens interpretadas permitem testes mais rápidos, evitando a necessidade de compilar o código explicitamente;

Compiladas



Compiladas

- Na compilação, o arquivo inteiro com o código é transformado numa sequência de comandos de baixo nível específicos da máquina e um outro arquivo executável é gerado.
- Um mesmo código pode gerar diferentes comandos de baixo nível em diferentes máquinas, mas a funcionalidade não será afetada.
- O usuário nem fica sabendo! Ele se preocupa mais com a lógica em si, e não com a máquina.



Swift



Como deve ser uma linguagem de programação?

- Rigidez sintática
 - “Vocabulário limitado”
 - Construções bem definidas
- Rigidez semântica
 - Sem ambiguidades
 - Intenção deve ser expressa de forma exata
 - “Computador somente cumpre ordens”
- Linguagem natural não pode ser utilizada para construir algoritmos computacionais

02

Por que aprender sobre paradigmas de programação?

Por que aprender sobre paradigmas de programação?

- Cada paradigma surgiu de necessidades diferentes;
- Cada um apresenta maiores vantagens sobre os outros dentro do desenvolvimento de determinado sistema;
- Um paradigma pode oferecer técnicas apropriadas para uma aplicação específica.

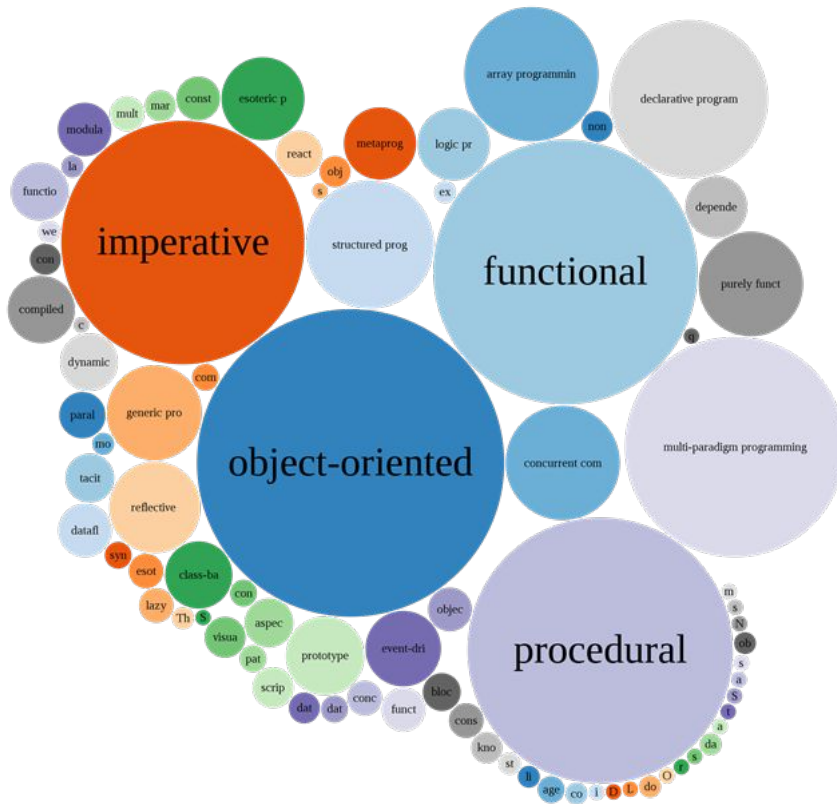
Por que aprender sobre paradigmas de programação?

- Com a escolha do paradigma de desenvolvimento adequado ao projeto, teremos alguns benefícios:
 - Produtividade;
 - Unicidade na orientação de escrita do código entre a equipe;
 - Legibilidade; e,
 - Facilidade de manutenção ao longo de sua existência.

03

Quais os principais paradigmas de programação?

Principaux Paradigmas



Por que aprender sobre paradigmas de programação?

- É um estilo de programação, caracterizado por uma seleção de conceitos
- Fornece a determina a visão que o programador possui sobre a estruturação e execução do programa
- Diferenciam-se pelas técnicas de programação que permitem ou proíbem
- Estão divididos em dois grupos/categorias:
 - Programação Imperativa
 - Programação Declarativa

Programação imperativa

Nesse tipo de construção, as instruções devem ser passadas ao computador na **sequência** em que devem ser executadas. Vários tipos de linguagem de programação suportam esse tipo de paradigma, como **Cobol**, **Fortran** e **Pascal**. O programador instrui a máquina sobre como devem ser computados os processos e como resolver um problema



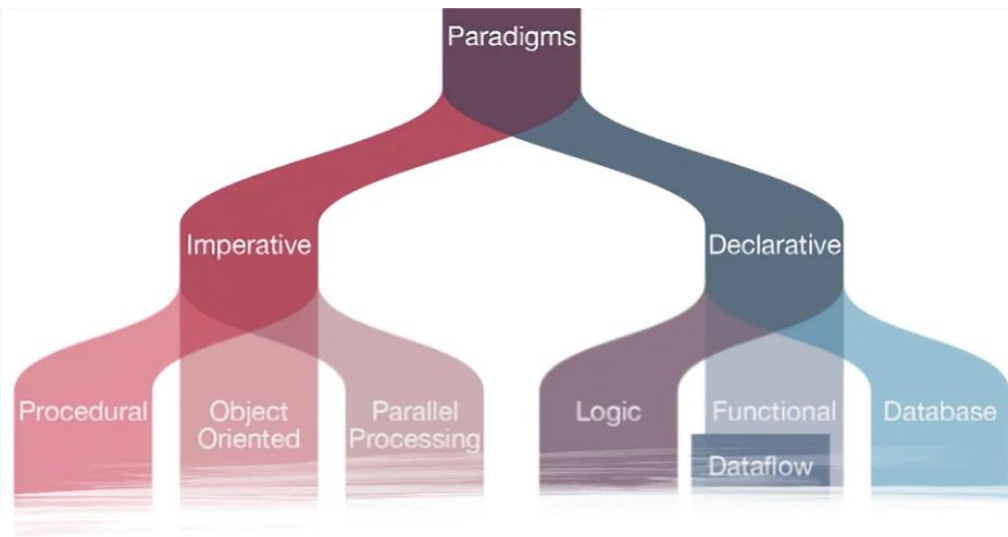
Programação imperativa

- Pontos importantes sobre essa programação:
 - Descreve o passo-a-passo dos procedimentos que a máquina deverá executar (daí o nome procedural);
 - A solução do problema é muito dependente da experiência e criatividade de quem trabalha com a programação;
 - O foco da resolução estará em “como” deve ser feito;
 - Ideal para projetos com poucas mudanças ao longo do tempo;
 - Eficiente e permite uma modelagem semelhante ao mundo real;
 - Bem estabelecido e flexível.

Programação declarativa

Nesse paradigma, há declarações iniciais de **verdades lógicas que são imutáveis**. Assim, depois de realizadas algumas interações entre elas, o resultado encontrado será sempre igual (para as declarações outrora feitas). Isso entra em contraste com a programação imperativa, na qual um mesmo trecho de código pode retornar resultados diferentes.

- Pontos importantes sobre essa programação:
 - Definem o que o programa faz e não como seus processos funcionam;
 - Ênfase nos resultados;




Source: <https://www.watelectronics.com/types-of-programming-languages-with-differences/>



Paradigma Procedural

Nesse tipo de programação, a pessoa passará uma espécie de passo-a-passo dos procedimentos que a máquina deverá executar (daí o nome procedural).

Nesse caso, a solução do problema será muito dependente da experiência e criatividade de quem trabalha com a programação. O foco da resolução estará em “como” deve ser feito. Características: **variáveis, comandos e procedimentos.**

An abstract graphic on the left side of the slide. It features several horizontal bars of varying lengths and colors (red, blue, green, yellow, black) stacked vertically. A large, light blue curly brace is positioned to the right of these bars, pointing towards the text. The background is a solid dark blue.

Paradigma Orientado a Objetos

Esse paradigma é bastante conhecido. Foi popularizado na década de 90 com a linguagem de programação **Java**, ao permitir uma programação multiplataforma. Antes disso, não era possível realizar tal tipo de trabalho.

Características: **objetos, métodos e classes.**



Paradigma Funcional

No paradigma de programação funcional, o uso de funções é destaque. O problema é dividido em blocos e, para sua resolução, são implementadas funções que definem variáveis em seu escopo e retornam algum resultado. São exemplos de linguagens suportadas por esse paradigma o **LISP**, o **Scheme** e o **Haskell**.

Paradigma funcional


- Pontos importantes sobre esse paradigma:
 - É bastante indicado quando a solução requerida **é fortemente dependente de uma base matemática**;
 - Subdivide-se o problema proposto e as funções implementadas farão os cálculos matemáticos;
 - Possui alocação de memória automática;
 - Características: **valores, expressões e funções**;



Paradigma lógico

O paradigma lógico é um tanto distinto dos demais paradigmas e deriva do declarativo. Fundamentalmente, utiliza formas de lógica simbólica como padrões de entrada e saída. A partir daí, realiza inferências para produzir os resultados.

Características: **asserções e relações.**



Paradigma orientado a eventos

O paradigma de orientação a eventos é usado por toda linguagem de programação que tem uso de recursos gráficos, como jogos e formulários. Dessa forma, a execução do programa se dá a medida que determinados eventos são disparados pelo usuário. Portanto, quem usa é responsável pelo momento em que o programa é executado.



04

Primeiros passos com a
linguagem python

O que é python?

- A linguagem de programação Python foi criada por Guido van Rossum em 1991;
- Ela é uma linguagem de alto nível interpretada que possui também orientação a objetos;
- Open Source;
- Código compreensível como o próprio inglês;
- Permite o rápido desenvolvimento (praticidade no uso);



Por onde começar?



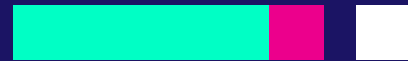
Aplicações web e IDE's



colab



Conceitos iniciais



Variáveis

- São pequenos espaços de memória, utilizados para armazenar e manipular dados;
- Em Python, os tipos de dados básicos são: tipo inteiro, float e tipo string;
- Cada variável pode armazenar apenas um tipo de dado a cada instante;
- Em Python, diferentemente de outras linguagens de programação, não é preciso declarar de que tipo será cada variável no início do programa.

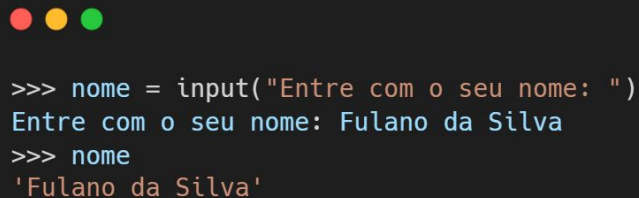
```
>>> a = 10
>>> a
10
```

```
>>> b = 1.2
>>> b
1.2
```

```
>>> c = "Olá Mundo"
>>> c
'Olá Mundo'
```

Variáveis


- A atribuição de valor para uma variável pode ser feita utilizando o comando `input()`, que solicita ao usuário o valor a ser atribuído à variável.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays a Python interactive session where a variable named 'nome' is assigned the value of the user input 'Fulano da Silva'.

```
>>> nome = input("Entre com o seu nome: ")
Entre com o seu nome: Fulano da Silva
>>> nome
'Fulano da Silva'
```

Variáveis

O comando `input()`, sempre vai retornar uma `string`. Nesse caso, para retornar dados do tipo inteiro ou `float`, é preciso converter o tipo do valor lido. Para isso, utiliza-se o `int (string)` para converter para o tipo inteiro, ou `float (string)` para converter para o tipo `float`.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays a Python interactive session where a variable 'num' is assigned the integer value 100.

```
>>> num = int(input("Entre com um numero? :"))
Entre com um numero? :100
>>> num
100
```

Variáveis

- Em Python, os nomes das variáveis devem ser iniciados com uma letra, mas podem possuir outros tipos de caracteres, como números e símbolos;
- O símbolo sublinha (`_`) também é aceito no início de nomes de variáveis;

Exemplo de nomes Variáveis

Nome	Válido	Comentários
a3	Sim	Embora contenha um número, o nome a3 inicia com letra.
velocidade	Sim	Nome formado com letras.
velocidade90	Sim	Nome formado por letras e números, mas inicia com letras.
salario_médio	Sim	O símbolo (_) é permitido e facilita a leitura de nomes grandes.
salario médio	Não	Nomes de variáveis não podem conter espaços em branco.
_salário	Sim	O sublinha (_) é aceito em nomes de variáveis, mesmo no início.
5A	Não	Nomes de variáveis não podem começar com números

Strings

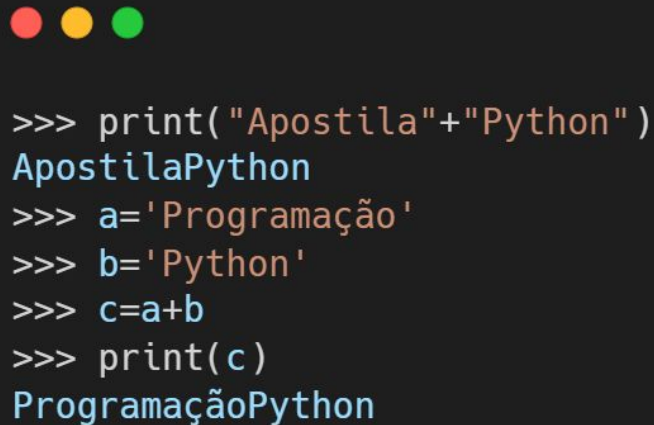
- Uma string é uma sequência de caracteres simples. Na linguagem Python, as strings são utilizadas com aspas simples ('... ') ou aspas duplas ("...");
- Para exibir uma string, utiliza-se o comando print();



```
>>> print("Olá mundo")
Olá mundo
>>>
```

Strings

- Para concatenar strings, utiliza-se o operador +



```
>>> print("Apostila"+"Python")
ApostilaPython
>>> a='Programação'
>>> b='Python'
>>> c=a+b
>>> print(c)
ProgramaçãoPython
```


Strings

Em Python, existem várias funções (métodos) para manipular strings. Na tabela a seguir são apresentados os principais métodos para a manipulação as strings.

Exemplo de manipulação de strings

Método	Descrição	Exemplo
len()	Retorna o tamanho da string.	teste = "Apostila de Python" len(teste) 18
capitalize()	Retorna a string com a primeira letra maiúscula	a = "python" a.capitalize() 'Python'
count()	Informa quantas vezes um caractere (ou uma sequência de caracteres) aparece na string.	b = "Linguagem Python" b.count("n") 2
startswith()	Verifica se uma string inicia com uma determinada sequência.	c = "Python" c.startswith("Py") True

Números

- Os quatro tipos numéricos simples, utilizados em Python, são números inteiros (int), números longos (long), números decimais (float) e números complexos (complex);
- A linguagem Python também possui operadores aritméticos, lógicos, de comparação e de bit;

Prática o/

The logo for Google Colab, featuring the word "colab" in a bold, lowercase, sans-serif font. The "co" is colored yellow, and the "lab" is colored orange. The logo is centered on a white rectangular background.

colab

Prática 2 o/

The logo for Google Colab, featuring the word "colab" in a bold, sans-serif font. The "co" is yellow with a subtle gradient, and "lab" is orange. The logo is centered on a white rectangular background.

colab

Obrigado!



CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.