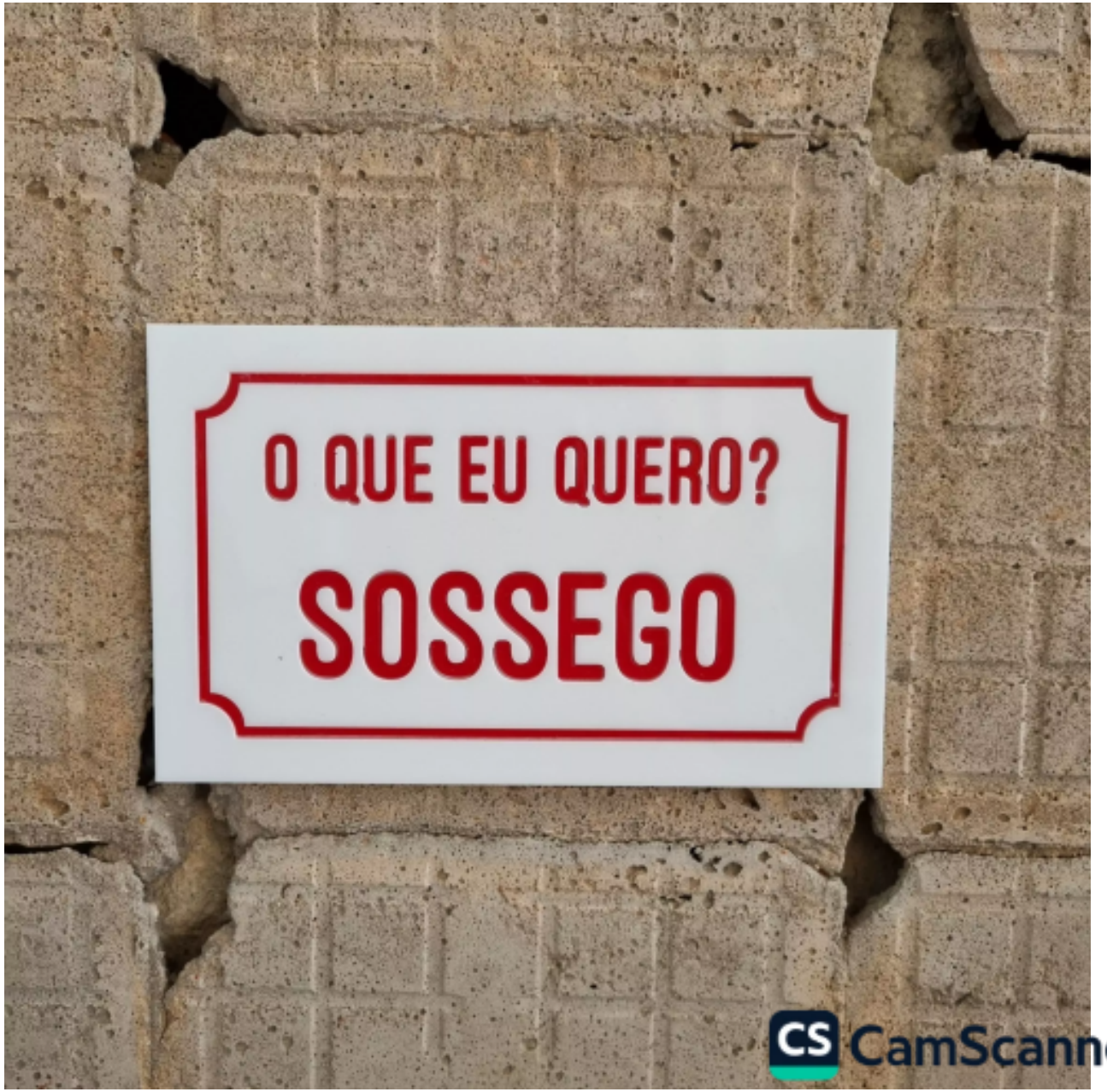
Introdução ao Python

Rodrigo Attique

Desenvolvedor de Sistemas

Prof. Rodrigo Attique

Python é vida!



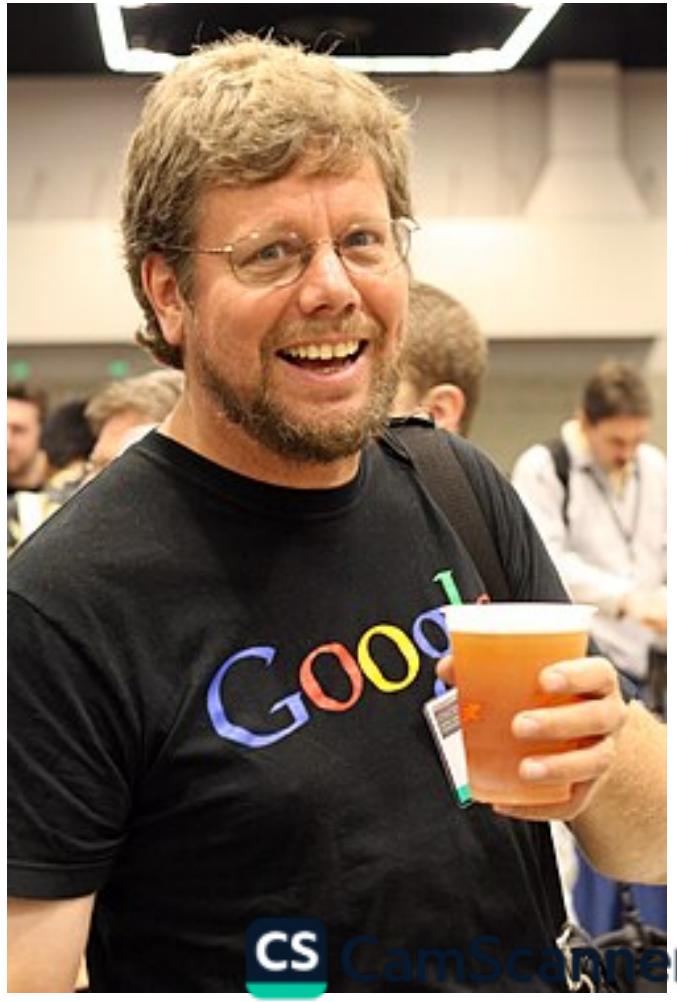
INTRODUÇÃO

Introdução

- **Python é uma linguagem de programação de alto nível,**
- **interpretada de script,**
- **imperativa,**
- **orientada a objetos,**
- **funcional,**
- **de tipagem dinâmica e forte**

Introdução

- Foi lançada por Guido van Rossum em 1991.
- Atualmente, possui um modelo de desenvolvimento comunitário, aberto e gerenciado pela organização sem fins lucrativos Python Software Foundation

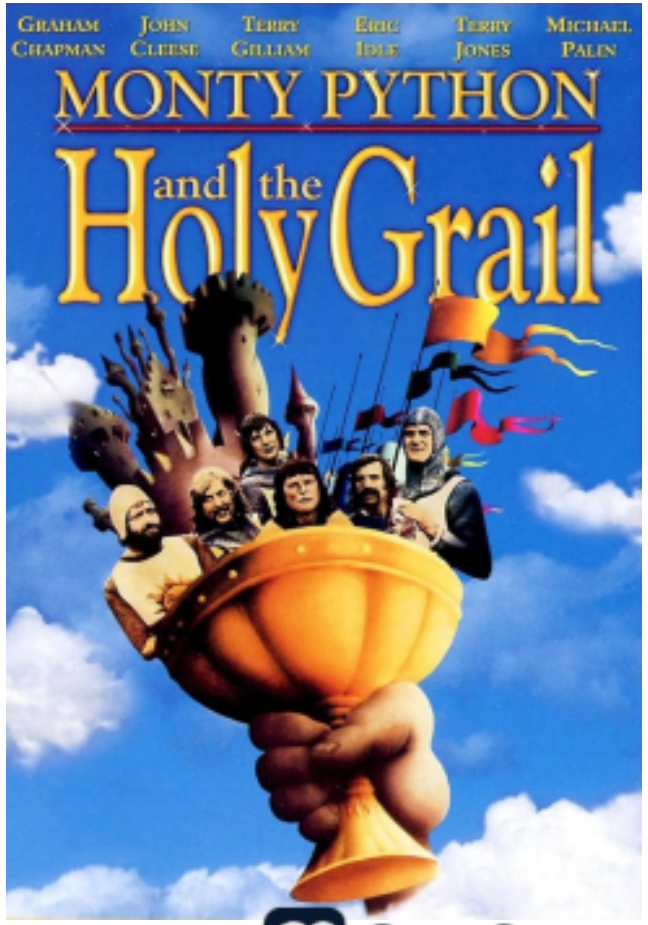


Introdução

- **A linguagem foi projetada com a filosofia de enfatizar a importância do esforço do programador sobre o esforço computacional.**
- **Prioriza a legibilidade do código sobre a velocidade ou expressividade.**
- **Combina uma sintaxe concisa e clara com os recursos poderosos de sua biblioteca padrão e por módulos e frameworks desenvolvidos por terceiros.**

Introdução

- **O nome Python teve a sua origem no grupo humorístico britânico Monty Python, criador do programa Monty Python's Flying Circus, embora muitas pessoas façam associação com o réptil do mesmo nome (em português, píton ou pitão).**



Introdução

Python foi feita com base na linguagem ABC, possui parte da sintaxe derivada do C, compreensão de listas, funções anonimas e função map de Haskell.

Os iteradores são baseados na Icon, tratamentos de exceção e módulos da Modula-3, expressões regulares de Perl.

Filosofia

Parte da cultura da linguagem gira ao redor de The Zen of Python, um poema que faz parte do documento "PEP 20 (The Zen of Python)",

escrito pelo programador em Python de longa data Tim Peters, descrevendo sumariamente a filosofia do Python. Entre os vinte princípios do poema, estão presentes:

Zen do Python

- **Bonito é melhor que feio;**
- **Explícito é melhor que implícito;**
- **Simples é melhor que complexo;**
- **Complexo é melhor que complicado;**
- **Legibilidade faz diferença.**

Pode-se vê-lo através de um *easter egg* do Python pelo comando:

```
>>> import this
```

Introdução

Atualmente, Python é um dos componentes padrão de vários sistemas operacionais, entre eles estão a maioria das distribuições do Linux, AmigaOS 4, FreeBSD, NetBSD, OpenBSD e OS X.

A linguagem se tornou a padrão no curso de ciências da computação do MIT em 2009;

Introdução

A linguagem também é usada em diversas áreas, como servidores de aplicação e computação gráfica.

Está disponível como linguagem de script em aplicações como OpenOffice (Python UNO Bridge), Blender e pode ser utilizada em procedimentos armazenados no sistema gerenciador de banco de dados PostgreSQL (PL/Python).

Introdução

Em 2012, foi criado o Raspberry Pi, cujo nome foi baseado na linguagem Python.

Uma das principais linguagens escolhidas é Python. Python influenciou várias linguagens, algumas delas foram Boo e Cobra, que usa a indentação como definição de bloco e Go, que se baseia nos princípios de desenvolvimento rápido de Python.

Desenvolvimento

O desenvolvimento de Python é conduzido amplamente através do processo Python Enhancement Proposal ("PEP"), em português Proposta de Melhoria do Python.

Os PEPs são documentos de projeto padronizados que fornecem informações gerais relacionadas ao Python, incluindo propostas, descrições, justificativas de projeto (design rationales) e explicações para características da linguagem.

PEPs pendentes são revisados e comentados por Van Rossum, o Benevolent Dictator for Life (líder arquiteto da linguagem) do projeto Python.

Módulos e frameworks

Ao longo do tempo têm sido desenvolvidos pela comunidade de programadores muitas bibliotecas de funções especializadas (módulos) que permitem expandir as capacidades base da linguagem.

Entre estes módulos especializados destacam-se:

Frameworks WEB

- Django – framework MVC
- TurboGears – framework MVC; usa CherryPy
- web2py – framework MVC inspirado no Django
- FastAPI – framework REST
- Flask – framework REST
- Plone – sistema de gerenciamento de conteúdo (CMS)
- Jam.py – framework low-code

Computação Gráfica

- PIL & Pillow – processamento de imagens
- PyOpenGL – suporte multiplataforma ao OpenGL
- Pygame – conjunto de módulos para o desenvolvimento de jogos eletrônicos com SDL

Computação científica e IA

- NumPy – matrizes e matemática
- pandas – análise de dados
- Matplotlib – biblioteca para manipulação de gráficos;
- TensorFlow – framework para aprendizado de máquina (ML)
- PyTorch – framework para aprendizado de máquina (ML)

Rede e bancos de dados

- **Twisted – framework orientado a eventos**
- **ZODB – sistema de persistência e banco de dados orientado a objetos**
- **SQLAlchemy – mapeamento objeto-relacional (ORM)**
- **SQLObject – mapeamento objeto-relacional (ORM)**

Interfaces gráficas

Tkinter – módulo padrão do Python; usa Tcl
PyGTK – interface para a biblioteca GTK
PyQt – interface para a biblioteca Qt
wxPython – interface para a biblioteca wxWidgets
Kivy – framework multiplataforma

Fontes

- https://pt.wikipedia.org/wiki/Python
- https://www.python.org/
- https://docs.python.org/pt-br/3.14/tutorial/introduction.html

PROGRAMANDO EM PYTHON

https://docs.python.org/pt-br/3.14/tutorial/index.html

ABRINDO O APETITE

Considerações iniciais - Interpretador

Python é interpretado: ou seja, o código é executado em tempo de execução por um interpretador que lê e executa linha a linha.

Isso torna o desenvolvimento extremamente rápido, mas pode deixar o código lento para tarefas muito críticas.

NOTA: Todas as palavras reservadas devem estar em minúsculo.

Considerações iniciais - Blocos de código

Diferente de linguagens C-Like que usam { }, chaves, para delimitar blocos de código.

Python usa indentação, ou tabulação, para delimitar um bloco de código.

Essa sintaxe é muito similar ao português ou linguagens humanas:

Dessa forma

E desa forma

Blocos de código

```
>>> x = int(input("Insira um número inteiro: "))
Insira um número inteiro: 42
>>> if x < 0:
...     x = 0
...     print('Negativo alterado para zero')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Um')
... else:
...     print('Mais')
...
Mais
```

```
>>> # Sequência de Fibonacci:
>>> # a soma de dois elementos define a próxima
>>> a, b = 0, 1
>>> while a < 10:
...     print(a)
...     a, b = b, a+b
```

Considerações iniciais - Tipos de Dados

Python usa tipagem dinâmica forte, ou seja, não é necessário especificar os tipos de dados ou declarar variáveis.

Tudo isso é feito diretamente pelo interpretador.

Em Python apenas inicializamos variáveis com o tipo desejado.

```
# este é o primeiro comentário
spam = 1  # e este é o segundo comentário
          # ... e agora um terceiro!
texto = "# Este não é um comentário por estar entre aspas."
```

Considerações iniciais - Comentários

Python utiliza #, cerquilha para comentários de uma linha;

- O termo não é HashTAG, o nome correto para o símbolo no inglês é simplesmente HASH, o termo hashtag é uma palavra combinada.

```
# este é o primeiro comentário
spam = 1  # e este é o segundo comentário
          # ... e agora um terceiro!
texto = "# Este não é um comentário por estar entre aspas."
```

Usando o shell interativo

É possível usar o shell interativo digitando python ou python3 em qualquer prompt de comando.

```
usuario@usuario-Latitude-3420:~$ python3
Python 3.12.3 (main, Jan 22 2026, 20:57:42) [GCC 13.3.0]
Type "help", "copyright", "credits" or "license" for mor
>>>
```

Shell Interativo

Isso nos permite simular ou testar instruções de código fora do programa principal.

Tipos de Número

Os números inteiros (por exemplo, 2, 4 e 20) são do tipo int, aqueles com parte fracionária (por exemplo, 5.0 e 1.6) são do tipo float. Veremos mais sobre tipos numéricos posteriormente neste tutorial.

Divisão (/) sempre retorna ponto flutuante (float). Para fazer uma divisão pelo piso e receber um inteiro como resultado você pode usar o operador //; para calcular o resto você pode usar o %:

Operador ** (asterisco-asterisco)

Com Python, é possível usar o operador ** para calcular potências [1]:

Atribuindo variáveis

O sinal de igual ('=') é usado para atribuir um valor a uma variável. Depois de uma atribuição, nenhum resultado é exibido antes do próximo prompt:

Mensagens de erro

Se uma variável não é “definida” (não tem um valor atribuído), tentar utilizá-la levantará um erro:

Ponto Flutuante e número com vírgula

Há suporte completo para ponto flutuante (float); operadores com operandos de diferentes tipos convertem o inteiro para ponto flutuante:

NOTA: Não use virgula para números reais, use ponto.

Outros números

Além de int e float, o Python oferece suporte a outros tipos de números, tais como Decimal e Fraction.

O Python também possui suporte embutido a números complexos, e usa os sufixos j ou J para indicar a parte imaginária (por exemplo, 3+5j).

Exercícios

1. Faça um Programa que peça um número e então mostre a mensagem O número informado foi [número].
2. Faça um Programa que peça dois números e imprima a soma.
3. Faça um Programa que peça as 4 notas bimestrais e mostre a média.
4. Faça um Programa que calcule a área de um quadrado, em seguida mostre o dobro desta área para o usuário.

Textos e Strings

Texto

Python pode manipular texto (representado pelo tipo str, também chamado de “strings”), bem como números. Isso inclui caracteres “!”, palavras “coelho”, nomes “Paris”, frases “Eu te protejo.”, etc. “Oba! :)”.

Texto

Eles podem ser colocados entre aspas simples ('...') ou aspas duplas ("...") com o mesmo resultado [2].

Aspas entre aspas

Para colocar aspas entre aspas, precisamos “escapá-la”, precedendo-as com \. Alternativamente, podemos usar o outro tipo de aspas:

Strings Literais

As strings literais podem abranger várias linhas. Uma maneira é usar as aspas triplas: """..."""" ou '''...'''.

Caracteres de fim de linha são incluídos automaticamente na string, mas é possível evitar isso adicionando uma \ no final. No exemplo a seguir, a nova linha no início não é incluída:

Strings Literais

Concatenando Strings

Concatenando Strings

Esse recurso é particularmente útil quando você quer quebrar strings longas:

Concatenando Strings

Isso só funciona com duas strings literais, não com variáveis ou expressões:

Concatenando Strings

Indexando Strings

As strings podem ser indexadas (subscritas), com o primeiro caractere como índice 0. Não existe um tipo específico para caracteres; um caractere é simplesmente uma string cujo tamanho é 1:

Indexando Strings

Indexando Strings

Indexando Strings

Indexando Strings

Indexando Strings

Uma maneira de lembrar como fatias funcionam é pensar que os índices indicam posições entre caracteres, onde a borda esquerda do primeiro caractere é 0. Assim, a borda direita do último caractere de uma string de comprimento n tem índice n, por exemplo:

Indexando Strings

A primeira fileira de números indica a posição dos índices 0...6 na string; a segunda fileira indica a posição dos respectivos índices negativos. Uma fatia de i a j consiste em todos os caracteres entre as bordas i e j, respectivamente.

Para índices positivos, o comprimento da fatia é a diferença entre os índices, se ambos estão dentro dos limites da string. Por exemplo, o comprimento de word[1:3] é 2.

Indexando Strings

Indexando Strings

No entanto, os índices de fatiamento fora do alcance são tratados graciosamente (N.d.T: o termo original “gracefully” indica robustez no tratamento de erros) quando usados para fatiamento. Um índice maior que o comprimento é trocado pelo comprimento, um limite superior menor que o limite inferior produz uma string vazia:

Alterando Strings

Alterando Strings

Exercícios

1. Escreva um programa que receba uma frase do utilizador e imprima o número total de caracteres (incluindo espaços).
2. Crie um programa que peça ao utilizador uma palavra e a exiba de três formas:
   a. Totalmente em maiúsculas.
   b. Totalmente em minúsculas.
   c. Apenas com a primeira letra em maiúscula.
3. Escreva um programa que peça uma frase e, em seguida, peça uma única letra. O programa deve dizer quantas vezes essa letra aparece na frase informada.

Exercícios

1. Crie um programa que receba o nome completo de uma pessoa e o exiba de trás para frente (invertido).

1. Escreva um programa que tenha uma frase fixa (ex: "Eu gosto de programar em Java"). Peça ao utilizador para digitar uma nova linguagem e substitua "Java" pelo valor digitado, exibindo a nova frase.

Listas

Listas

Python inclui diversas estruturas de dados compostas, usadas para agrupar outros valores.

A mais versátil é list (lista), que pode ser escrita como uma lista de valores (itens) separados por vírgula, entre colchetes.

Os valores contidos na lista não precisam ser todos do mesmo tipo.

Listas

Concatenando Listas

Alterando Listas

Alterando Listas

Alterando Listas

Fatiando Listas

Fatiando Listas

Contando Listas

Aninhar Listas

Controle de Fluxo

Condições - IF/ELSE/ELIF

Bloco IF

Instrução IF (se/senão)

Pode haver zero ou mais partes elif, e a parte else é opcional. A palavra-chave ‘elif’ é uma abreviação para ‘else if’, e é útil para evitar indentação excessiva. Uma sequência if … elif … elif … substitui as instruções switch ou case, encontrados em outras linguagens.

Se você está comparando o mesmo valor com várias constantes, ou verificando por tipos ou atributos específicos, você também pode achar a instrução match útil. Para mais detalhes veja Instruções match.

Instrução FOR

Instrução FOR

A instrução for em Python é um pouco diferente do que costuma ser em C ou Pascal.

Ao invés de sempre iterar sobre uma progressão aritmética de números (como no Pascal), ou permitir ao usuário definir o passo de iteração e a condição de parada (como C), a instrução for do Python itera sobre os itens de qualquer sequência (seja uma lista ou uma string), na ordem que aparecem na sequência.

Por exemplo:

Exemplo FOR

Código que modifica uma coleção sobre a qual está iterando pode ser inseguro.

No lugar disso, usualmente você deve iterar sobre uma cópia da coleção ou criar uma nova coleção:

Código que modifica uma coleção

A função range()

Se você precisa iterar sobre sequências numéricas, a função embutida range() é a resposta. Ela gera progressões aritméticas:

A função range()

O ponto de parada fornecido nunca é incluído na lista; range(10) gera uma lista com 10 valores, exatamente os índices válidos para uma sequência de comprimento 10. É possível iniciar o intervalo com outro número, ou alterar a razão da progressão (inclusive com passo negativo):

Iterando sobre indices

Função enumerate()

Curiosidades sobre o range()

Em muitos aspectos, o objeto retornado pela função range() se comporta como se fosse uma lista, mas na verdade não é. É um objeto que retorna os itens sucessivos da sequência desejada quando você itera sobre a mesma, mas na verdade ele não gera a lista, economizando espaço.

Dizemos que um objeto é iterável, isso é, candidato a ser alvo de uma função ou construção que espera alguma coisa capaz de retornar sucessivamente seus elementos um de cada vez. Nós vimos que a instrução for é um exemplo de construção, enquanto que um exemplo de função que recebe um iterável é sum():

Instruções break e continue

Instruções break e continue

Cláusulas else em laços

Em um laço for ou while a instrução break pode ser pareada com uma cláusula else. Se o laço terminar sem executar o break, a cláusula else será executada.

Em um laço for, a cláusula else é executada após o laço finalizar sua iteração final, ou seja, se não ocorrer nenhuma interrupção.

Cláusulas else em laços

Em um laço while, ele é executado após a condição do laço se tornar falsa.

Em qualquer tipo de laço, a cláusula else não é executada se o laço foi encerrado por um break.

Claro, outras maneiras de encerrar o laço mais cedo, como um return ou uma exceção levantada, também pularam a execução da cláusula else.

Cláusulas else em laços

(Sim, este é o código correto. Observe atentamente: a cláusula else pertence ao laço for, não à instrução if.)

Uma maneira de pensar na cláusula else é imaginá-la pareada com o if dentro do laço.

Conforme o laço é executado, ele executará uma sequência como if/if/if/else.

Cláusulas else em laços

O if está dentro do laço, encontrado várias vezes. Se a condição for verdadeira, um break acontecerá. Se a condição nunca for verdadeira, a cláusula else fora do laço será executada.

Cláusulas else em laços

Quando usado em um laço, a cláusula else tem mais em comum com a cláusula else de uma instrução try do que com a de instruções if: a cláusula else de uma instrução try é executada quando não ocorre exceção, e a cláusula else de um laço é executada quando não ocorre um break.

Para mais informações sobre instrução try e exceções, veja Tratamento de exceções.

Instrução PASS

Instruções pass

Instruções pass

Instrução pass

Instrução pass

Outra ocasião em que o pass pode ser usado é como um substituto temporário para uma função ou bloco condicional, quando se está trabalhando com código novo, ainda indefinido, permitindo que mantenha-se o pensamento num nível mais abstrato. **O pass é silenciosamente ignorado:**

Instrução pass

Para este último caso, muitas pessoas usam o literal de reticências ... em vez de pass.

Este uso não tem significado especial para Python e não faz parte da definição da linguagem (você pode usar qualquer expressão constante aqui), mas ... também é usado convencionalmente como um espaço reservado.

Consulte O Objeto Ellipsis.

Instruções match

Instruções match

Uma instrução match pega uma expressão e compara seu valor com padrões sucessivos fornecidos como um ou mais blocos de case. Isso é superficialmente semelhante a uma instrução switch em C, Java ou JavaScript (e muitas outras linguagens), mas muito mais parecido com a correspondência de padrões em linguages como Rust ou Haskell.

Apenas o primeiro padrão que corresponder será executado, podendo também extrair componentes (elementos de sequência ou atributos de objetos) do valor para variáveis. Se nenhum caso corresponder, nenhuma das ramificações será executada.

Instruções match

Instruções match

Pronto!

Ainda há muito o que aprender sobre Python, para nos aperfeiçoar é fundamental a prática de exercícios e situações que nos permitam evoluir.

Documentação de referência

https://docs.python.org/pt-br/3.14/tutorial

Prof. Rodrigo Attique

Obrigado!