Python - Monitoria 2024 - Aula 1

```
print("Bem-Vindos à Oficina de Python 2024")
```

Nossos Monitores

Gabriel

Github
Linkedin

Leo

Github
Linkedin

Lucas

Github
Linkedin

Giovanni

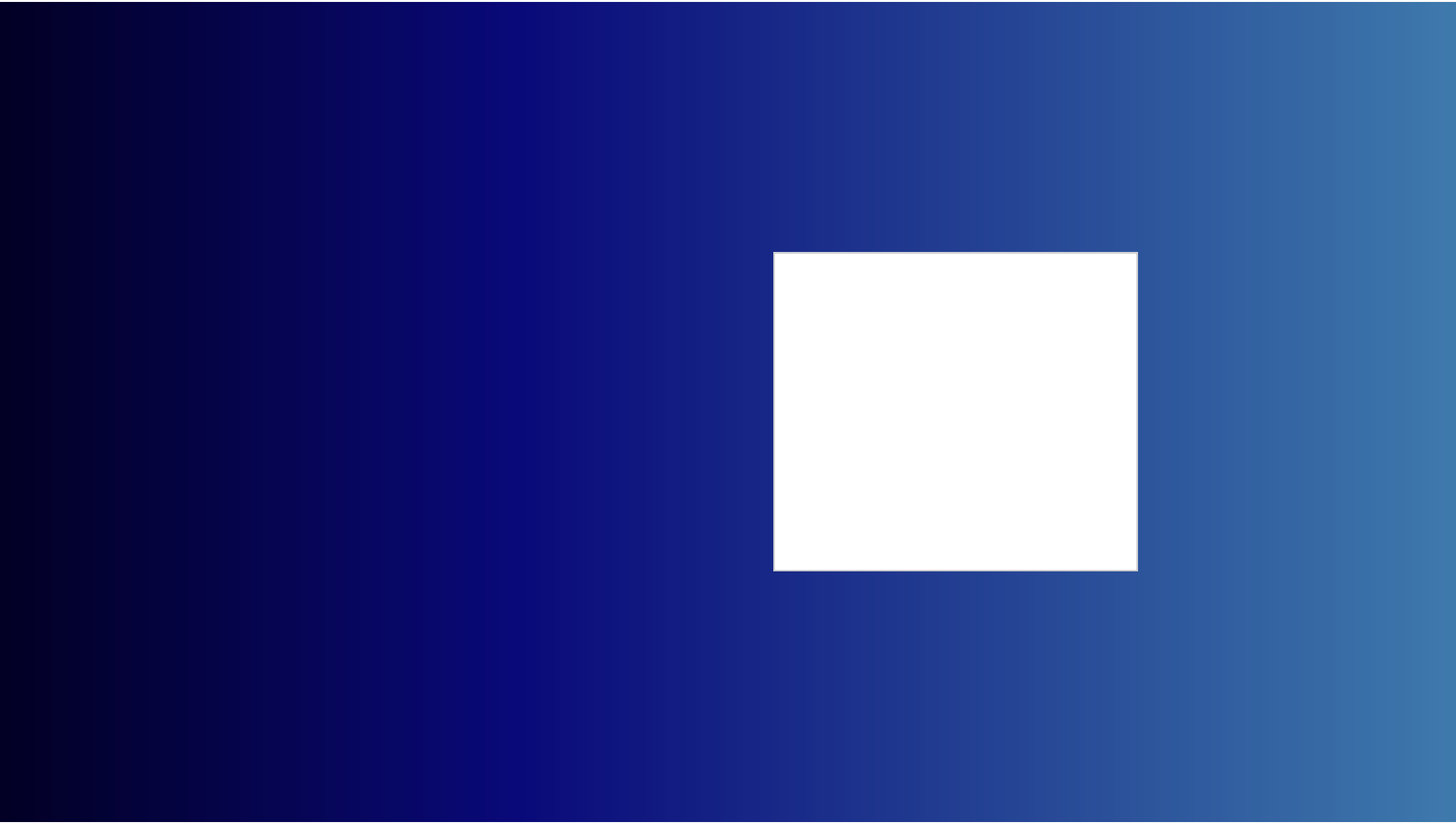
Github
Linkedin

Sumário do Conteúdo

Código da Sala Online: *a4lb5gi*

Link de Convite


Alt text



História do Python

Python é uma linguagem de programação criada por **Guido van Rossum** e lançada em 1991. Seu nome foi inspirado no grupo de comédia britânico **Monty Python**. Python foi projetado para enfatizar a legibilidade do código, permitindo que programadores expressem conceitos em menos linhas de código comparado a outras linguagens como C++ ou Java.

- **Contexto Histórico**: Na época de sua criação, Guido van Rossum estava procurando por uma forma de hobby para ocupar seu tempo na semana entre o Natal e o Ano Novo. Ele decidiu escrever um interpretador para uma nova linguagem de script, e assim nasceu o Python.
- Primeira Versão: A versão 0.9.0 de Python foi lançada em fevereiro de 1991 e já incluía muitas das principais características que associamos ao Python moderno: classes com herança, tratamento de exceções, e as funções.

Por que Python foi criado?

Python foi desenvolvido com a intenção de ser uma linguagem simples e intuitiva, acessível para iniciantes, mas também poderosa para desenvolvedores experientes. Guido van Rossum se inspirou em linguagens como ABC, mas queria algo mais expansível e aplicável a uma maior variedade de tarefas.

Principais Objetivos

- **Simplicidade**: Python deveria ser fácil de ler e escrever.
- **Extensibilidade**: Deveria ser possível ampliar a linguagem através de módulos e pacotes.
- **Portabilidade**: O código Python deveria rodar em diversas plataformas sem modificações.

Usabilidade e Curiosidades

- **Versatilidade**: Utilizada em diversas áreas como desenvolvimento web, automação, ciência de dados, inteligência artificial, entre outros.
- **Comunidade Ativa**: Grande suporte da comunidade e vasto ecossistema de bibliotecas. Python possui um dos maiores repositórios de pacotes, conhecido como PyPI (Python Package Index), que contém mais de 300.000 pacotes.
- **Curiosidade**: Python é frequentemente classificada entre as linguagens mais populares e mais ensinadas em universidades ao redor do mundo. Segundo o IEEE Spectrum, Python tem sido consistentemente classificada como a linguagem de programação mais popular do mundo.
- **Python na Educação**: Python é amplamente usado como a primeira linguagem de programação ensinada nas universidades devido à sua sintaxe clara e o suporte a múltiplos paradigmas, como a programação procedural e orientada a objetos.

Características da Linguagem

Linguagem interpretada

O fato de ser uma **linguagem interpretada**, o que significa que ela não precisa passar pelo processo de *compilação*.

- O processo de interpretação é executado dentro de **máquinas virtuais**, nas quais o código passa por uma camada intermediária que irá traduzir os comandos de programa para *código binário*.

Isso acelera bastante a velocidade de desenvolvimento.

Sintaxe simples

Sua sintaxe é simples, fácil de aprender e muito próxima da linguagem falada por nós.

Por isso, podemos dizer que ela se trata de uma linguagem de alto nível.

Multiparadigma

Ela é um **multiparadigma**, pois nos dá a possibilidade de programar em vários paradigmas, tais como:

- **Procedural**, com instruções transmitidas ao computador na sequência em que devem ser executadas;
- **Funcional**, paradigma que consiste em programas construídos aplicando e compondo funções;
- **Orientação a objetos**, que traz a perspectiva do mundo real para a programação, tornando os programas fáceis de entender devido a essa relação.

O próprio programa “reconhece” qual tipo de dado está sendo utilizado, fazendo com que ele não precise ser previamente declarado. Por isso dizemos que ele possui uma semântica dinâmica.

O Python é utilizado para desenvolvimento de software, análise de dados, inteligência artificial, automação de tarefas, criação de aplicativos web e daí por diante.

Ele também possui uma vasta biblioteca padrão e uma comunidade ativa de pessoas desenvolvedoras, o que facilita encontrar soluções e recursos para diferentes projetos.

As áreas que mais utilizam Python são:

- A análise de dados
- Aprendizado de máquina
- Desenvolvimento web
- DevOps

- Créditos: Python Developers Survey 2021 Results

Por que aprender Python?

- Possui uma Sintaxe simples;
- É Multiplataforma e de código aberto;
- É versátil;
- Tem uma comunidade fiel e ativa;
- É utilizado por grandes empresas;
- É o mais popular na Ciência de Dados;
- Está em alta no mercado de trabalho.

Como instalar python no windows?

- Para instalar o Python no seu sistema operacional **Windows**, você deve baixar o instalador disponível na página de download oficial do Python e clicar em download, como mostrado abaixo.

Faça o download do instalador executável do Windows e clique duas vezes para iniciar o assistente de instalação.

O processo de instalação é bem simples:

- Marque a opção “Add Python to PATH”;
- Clique em “Install Now”;
- Após a instalação, basta clicar no botão “Close”;
- Para verificar se a instalação do Python foi bem-sucedida, pesquise no Menu Iniciar por “cmd” e clique duas vezes para abri-lo;

- Digite o comando `python --version` para se assegurar da versão do Python que está instalada.

- Agora digite `pip --version.` Este comando retornará a versão do pip instalada em sua máquina. O pip é o gerenciador de pacotes. Com ele, você poderá adicionar novas funcionalidades ao Python.

Quais são os tipos de dados em Python?

Como você já sabe, Python é uma linguagem **dinamicamente tipada**, o que significa que não é necessário declarar o tipo de variável ou fazer casting (mudar o tipo de variável), pois o Interpretador se encarrega disso para nós.

Isso significa também que, se por algum motivo precisarmos **alterar o tipo de variável durante a execução do programa**, é possível fazer essa mudança.

Tipos de Dados

- Inteiro (**int**); Exemplo: `1`
- Ponto Flutuante ou Decimal (**float**); Exemplo: `1.1`
- Tipo Complexo (**complex**); Exemplo: `8j`
- String (**str**); Exemplo: `"hello"`
- Boolean (**bool**); Exemplo: `true / false`
- List (**list**); Exemplo: `['Mônica', 'Ana', 'Bruno', 'Alice']`
- Tuple; Exemplo: `(90, 79, 54, 32, 21)`
- Dictionary (dic); Exemplo: `{'Camila': 1.65, 'Larissa': 1.60, 'Guilherme': 1.70}`

01) Tipo Inteiro (int)

É um tipo usado para um número que pode ser escrito sem um componente decimal, podendo ser positivo ou negativo.

No código abaixo, por exemplo, vemos as variáveis, idade e ano, com os valores 20 e 2010 atribuídos a elas; se pedirmos que o programa imprima o tipo das variáveis, teremos como retorno que elas são do tipo inteiro:

```
idade = 20
ano = 2010

print(type(idade))
print(type(ano))

<class ‘int’>
<class ‘int’>
```

02) Ponto Flutuante ou Decimal (float)

É um tipo composto por números decimais. O float é usado para números racionais (que podem ser representados por uma fração), informalmente conhecidos como “números quebrados”.

No código abaixo, por exemplo, vemos as variáveis altura e peso com os valores 1.73 e 78.500 atribuídos a elas. Se pedirmos que o programa imprima o tipo das variáveis, teremos como retorno que elas são do tipo float:

```
altura = 1.73
peso = 78.500

print(type(peso))
print(type(altura))

<class ‘float’>
<class ‘float’>
```

03) String (str)

É um conjunto de caracteres geralmente utilizados para representar palavras, frases ou textos.

Temos como exemplo as variáveis nome e profissão, com os dados Guilherme e Engenheiro de Software atribuídos a elas.

Pedindo para o programa imprimir o tipo dessas variáveis, teremos como retorno que elas são strings:

```
nome = ‘Guilherme’
profissao = ‘Engenheiro de Software’

print(type(profissao ))
print(type(nome))

<class ‘str’>
<class ‘str’>
```

04) Boolean (bool)

Tipo de dado lógico que pode representar apenas dois valores: falso ou verdadeiro (False ou True, em Python).

Na lógica computacional, podem ser considerados como 0 ou 1. Como exemplo, temos as variáveis: sexta_feira e feriado, com os dados True e False atribuídos a elas.

Se pedirmos que o programa imprima o tipo das variáveis, vamos ter como retorno que elas são do tipo boolean:

```
sexta_feira = True
feriado = False

print(type(sexta_feira))
print(type(feriado))

<class 'bool'>
<class 'bool'>
```

Como mudar o tipo da variável no Python?

Em alguns cenários pode ser necessário mudar o tipo de uma variável e no Python isso é muito fácil, justamente por se tratar de uma linguagem dinamicamente tipada.

Vejamos a seguir, alguns exemplos de como trocar os tipos de variáveis.

Float para String

```
#Antes de converter
altura=1.55
print(type(altura))

#Fazendo a conversão
altura = str(altura)

#Depois da conversão
print(type(altura))
print(altura)

<class ‘float’>
<class ‘str’>
 1.55
```

Inteiro pra Float

```
#Antes de converter
idade=18
print(type(idade))

#Fazendo a conversão
idade = float(idade)

#Depois da conversão
print(type(idade))
print(idade)

<class ‘intt’>
<class ‘float’>
18.0
```

Operadores em Python

Os operadores são usados na construção de expressões, as quais contém um número variado de operandos. Por exemplo, na expressão `a + b`, temos o operador de aritmético `+` e operandos são as variáveis `a e b`.

Na linguagem Python temos a seguinte separação entre os diferentes tipos de operadores:

- Operadores aritméticos
- Operadores de atribuição
- Operadores de comparação
- Operadores lógicos
- Operadores de identidade
- Operadores de associação

Operadores Aritméticos

Os operadores aritméticos são utilizados na execução de operações matemáticas, tais como a soma e a subtração, por exemplo.

```python
numero_1 = 5
numero_2 = 2

soma = numero_1 + numero_2
subtracao = numero_1 - numero_2
multiplicacao = numero_1 * numero_2
divisao = numero_1 / numero_2
divisao_inteira = numero_1 // numero_2
modulo = numero_1 % numero_2
exponenciacao = numero_1 ** numero_2

print(soma) # 7
print(subtracao) # 3
print(multiplicacao)  # 10
print(divisao) # 2.5
print(divisao_inteira) # 2
print(modulo)  # 1
print(exponenciacao) # 25
```

Uma característica importante a ser observada quando falamos dos operadores matemáticos é a precedência. Essa característica é relativa à ordem da execução deles e acontece quando mais de um operador está presente numa expressão. Segue a precedência dos operadores no Python.

- As expressões contidas em parênteses têm a precedência maior na linguagem Python. Isso permite que uma expressão execute antes de outra. Ex.:

```
print((2 + 5) * 3) # O resultado será 21
```

- Após os parênteses, o próximo operador com maior precedência é o de exponenciação. Ex.:

```
print( 1 + 5**2 ) # O resultado será 26
```

- Multiplicação e divisão têm precedência sobre a adição e subtração: como já é conhecido na matemática, divisão e multiplicação são executadas antes das operações de adição e subtração. Ex.:

```
print(5 * 3 + 8) # O resultado será 23
```

- Ordem de precedência é avaliada da esquerda para a direita. Portanto, após os operadores anteriores, a sequência da execução será da esquerda para a direita. Ex.:

```
print(8 + 5 - 10) # O resultado será 3
```

Operadores de atribuição

Os operadores de atribuição atribuem valor a uma variável.

```python
numero_1 = 5

numero_1 += 5

print(numero_1) # O resultado será 10
```

Operadores de comparação

Os operadores de comparação são usados para comparar valores, o que vai retornar `True ou False`, dependendo da condição.

Operadores lógicos

Os operadores lógicos são usados para unir duas ou mais expressões condicionais. Isso é feito por meio de conectivos.

Operadores de identidade

Os operadores de identidade servem para a comparação de objetos. Nessa comparação é verificado se eles ocupam a mesma posição na memória, o que significará que se trata do mesmo objeto.

Operadores de associação

Os operadores de associação são utilizados para verificar se uma sequência contém um objeto.

```
TODO: Remover esse slide
- Fazer slides sobre: Entrada de Dados (Input)
- Fazer exercícios para serem passados em sala
- Fazer exercício para casa Aula 1 20/08
```