



P Y T H O N & S U A S A P L I C A Ç Õ E S

FUNDAMENTOS E AMBIENTE DE DESENVOLVIMENTO

POR JOÃO VÍTOR PAMPONET ESTEVES

O QUE IREMOS APRENDER

1

Apresentação do Curso

2

Introdução à plataforma GitHub

3

Programação na Engenharia

4

Conceitos básicos de programação

5

Operadores Aritméticos

6

Introdução aos ambientes de desenvolvimento (IDE)

7

Variáveis e tipos de dados primitivos

8

Entrada e saída de dados no Python

OI, EU SOU O JOÃO!

Desenvolvedor com habilidades em:

- Desenvolvimento Web
- Análise de Dados
- Desenvolvimento de Apps
- Produção de Conteúdos Didáticos de Programação

Formação acadêmica

- Graduação em Análise e Desenvolvimento de Sistemas
- Pós-graduação em Segurança da Informação
- Curso de Programação Full Stack

**ACESSE MEU
PORTFÓLIO!**



joaopamponet.vercel.app

MINHA TRAJETÓRIA

2023 - Projeto Social NID

Por meio da ministração de aulas de informática básica, facilitei a inclusão digital de pessoas em situação de vulnerabilidade, abrindo novas oportunidades e acesso à informação.

2025 - Ford Brasil

Atualmente atuo no time de Inovação Aprendizado e desenvolvimento como pesquisador, realizando análise de dados e desenvolvimento de plataformas para uso interno das lideranças internacionais.

2024 - Infinity School

Como assistente acadêmico de programação, Implementei de ferramentas que simplificaram processos e desenvolvi conteúdos educacionais que potencializaram o aprendizado dos alunos em sala de aula.

APRESENTAÇÃO DO CURSO

Nesta primeira aula, vamos dar início ao curso Python e Suas Aplicações para Engenheiros.

O objetivo é apresentar conceitos fundamentais de programação e explorar como o Python pode ser utilizado como ferramenta de apoio na engenharia.

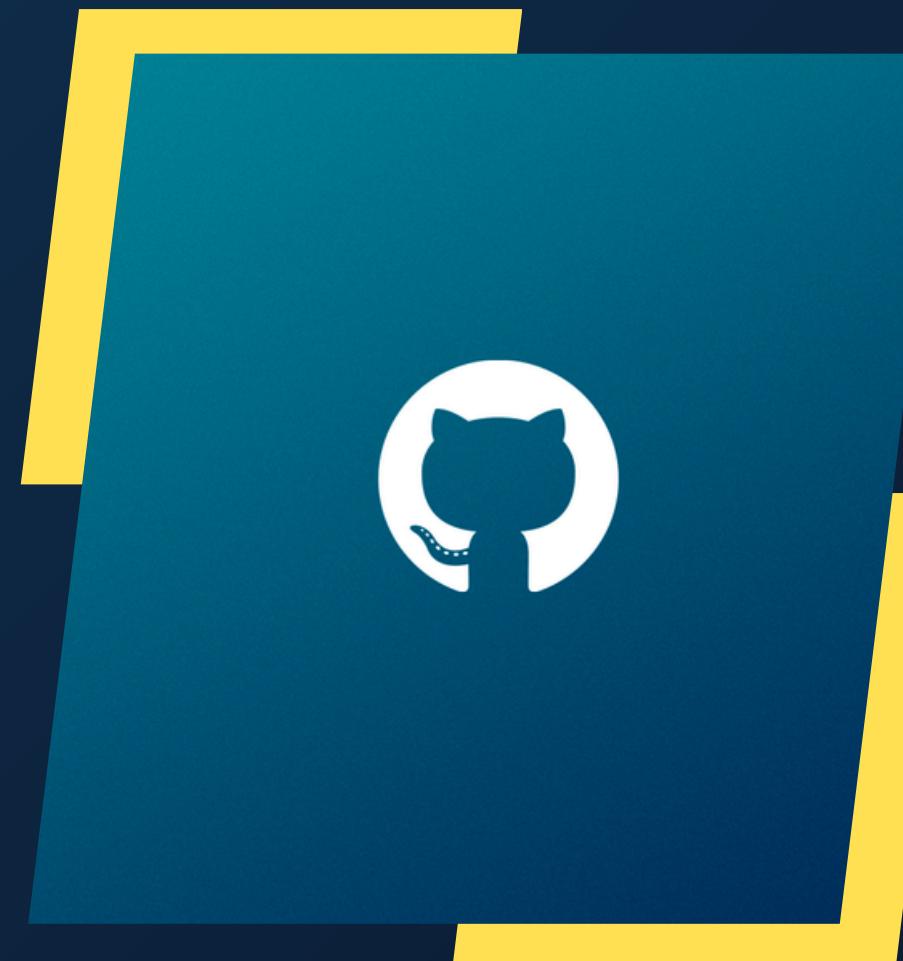
Durante as próximas aulas, trabalharemos desde os fundamentos da linguagem até aplicações práticas em cálculos, automação de tarefas e análise de dados.

A expectativa é que, ao final do curso, você seja capaz de desenvolver programas que resolvam problemas reais do cotidiano da engenharia.



```
31     self._logger = None
32     self.file = None
33     self.fingerprints = set()
34     self.logduplicates = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path,
39                         'seen'), 'a')
40         self.file.seek(0)
41
42     @classmethod
43     def from_settings(cls, settings):
44         debug = settings.getbool('DEBUG')
45         return cls(job_dir(settings), debug)
46
47     def request_seen(self, request):
48         fp = self.request_fingerprint(request)
49         if fp in self.fingerprints:
50             return True
51         self.fingerprints.add(fp)
52         if self.file:
53             self.file.write(fp + os.linesep)
54
55     def request_fingerprint(self, request):
56         return request_fingerprint(request)
```

INTRODUÇÃO À PLATAFORMA GITHUB



O GitHub é uma plataforma online que utiliza o sistema de controle de versão Git para armazenar, organizar e compartilhar códigos de forma colaborativa. Ele registra todas as alterações feitas nos arquivos, permitindo recuperar versões anteriores, acompanhar o histórico e trabalhar em equipe de qualquer lugar.

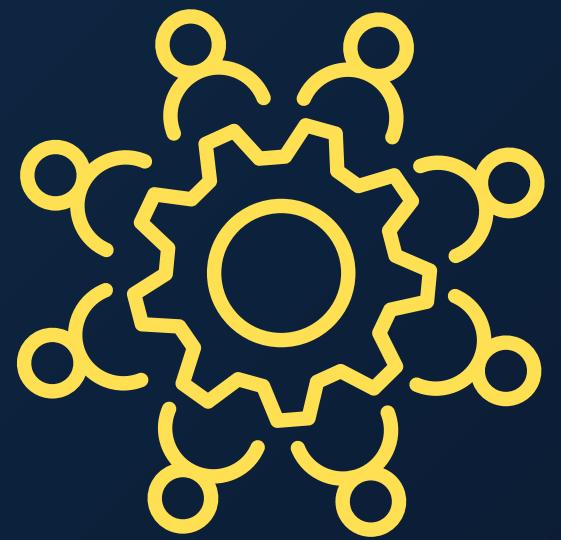
É uma ferramenta útil para documentar projetos, além de facilitar a colaboração de múltiplos desenvolvedores no projeto.

Neste curso, o GitHub servirá como central de materiais, reunindo exemplos de código, listas de exercícios e atualizações de cada aula em um único repositório.

PRINCIPAIS FUNCIONALIDADES DO GITHUB



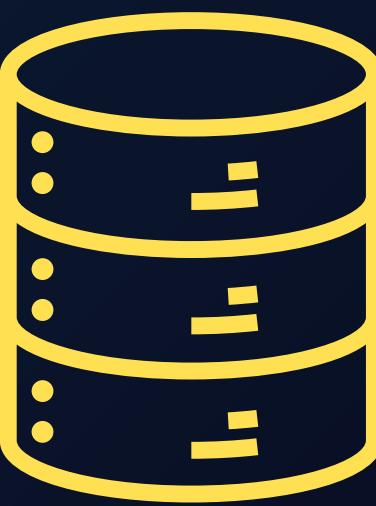
Controle de versão
Permite voltar no tempo e restaurar qualquer versão anterior do código.



Colaboração
Vários usuários podem trabalhar no mesmo projeto, cada um fazendo alterações que depois são integradas.



Armazenamento em nuvem
Seus códigos ficam disponíveis online, podendo ser acessados de qualquer computador conectado à internet.



Transparência e histórico
Todas as modificações ficam registradas, com autor, data e descrição das mudanças.

POR QUE O GITHUB É ÚTIL ?



Muitos projetos envolvem cálculos, simulações, relatórios e scripts que precisam ser constantemente atualizados. O GitHub facilita esse processo porque:

- Permite documentar cada passo do desenvolvimento de um programa ou modelo.
- Torna fácil compartilhar o código com colegas ou equipes de pesquisa.
- Garante que nenhuma informação seja perdida, já que há um histórico completo.
- Possibilita publicar projetos abertos, colaborando com a comunidade científica e profissional.

USO NO CURSO

No contexto deste curso, o GitHub será utilizado como central de materiais. Todos os exemplos de código, listas de exercícios e até as soluções de atividades serão disponibilizados em um repositório exclusivo. O aluno poderá:

Acessar o repositório

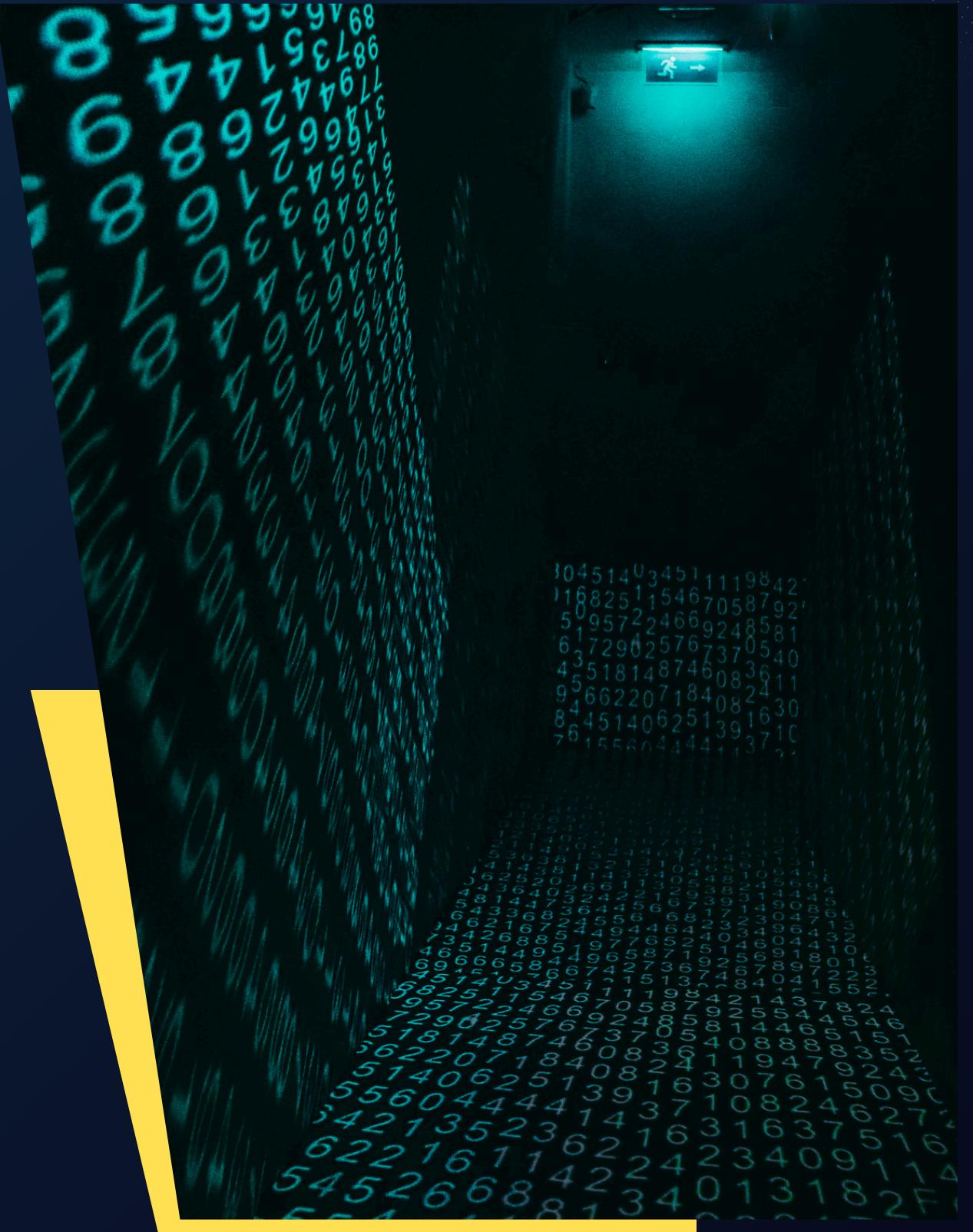
visualizar e baixar os arquivos para o seu computador.

Navegar entre pastas

localizar os conteúdos de cada aula.

Compartilhar Seus Projetos

A avaliação dos trabalhos do curso será realizada através do compartilhamento de repositórios na plataforma.





CONTEXTUALIZAÇÃO DA PROGRAMAÇÃO NA ENGENHARIA

Do cálculo estrutural à simulação de fenômenos físicos, do controle de processos industriais à análise de grandes volumes de dados, a automação proporcionada por linguagens como o Python traz ganhos de tempo, precisão e eficiência.

PROGRAMAÇÃO NA ENGENHARIA

Processos que antes exigiam cálculos manuais extensos ou softwares especializados podem ser automatizados com scripts em Python, economizando tempo e reduzindo a possibilidade de erros.





PROGRAMAÇÃO NA ENGENHARIA

A programação oferece ao engenheiro maior **autonomia**. Em vez de depender exclusivamente de softwares prontos e muitas vezes caros, é possível desenvolver soluções sob medida para cada **problema**. Isso torna o profissional mais versátil e preparado para lidar com situações em que não há ferramentas prontas disponíveis.

SOBRE PYTHON



Uma das linguagens de programação mais populares no mundo, e isso não é por acaso.

Ela combina uma sintaxe simples e intuitiva com um grande poder de aplicação prática. Sua escrita lembra o inglês, o que torna o código fácil de ler e compreender até por quem está começando.

Outro ponto de destaque é a vasta comunidade de desenvolvedores, que contribuem constantemente com bibliotecas e pacotes voltados para diferentes áreas: desde matemática e estatística até inteligência artificial, simulação numérica e automação de processos industriais.

BIBLIOTECAS

Para engenheiros, bibliotecas como NumPy (cálculo numérico), Pandas (análise de dados), Matplotlib (visualização gráfica) e SciPy (cálculos científicos) são ferramentas extremamente úteis.



CASES DE SUCESSO



TESLA

APLICAÇÃO

Desde o desenvolvimento de softwares embarcados para veículos elétricos até análise de dados em larga escala.

FERRAMENTAS

Bibliotecas como NumPy e Pandas são aplicadas no processamento e análise de grandes volumes de dados coletados dos sensores dos carros.

CASE

No desenvolvimento do Autopilot, o sistema de direção autônoma da Tesla, Python é essencial para lidar com dados captados por câmeras e radares, permitindo a interpretação do ambiente em tempo real.

CASES DE SUCESSO



APLICAÇÃO

A Airbus adota Python em processos de engenharia aeronáutica, especialmente na modelagem, simulação e análise de desempenho de aeronaves.

FERRAMENTAS

Bibliotecas como SciPy e Matplotlib auxiliam na simulação de fenômenos físicos e na análise de dados de testes.

CASE

Um dos destaques é o uso do Python no projeto de modelagem de sistemas de controle de aeronaves.



CASES DE SUCESSO

APLICAÇÃO

A Siemens utiliza Python para automação industrial, manutenção preditiva e análise de grandes volumes de dados em ambientes de produção.

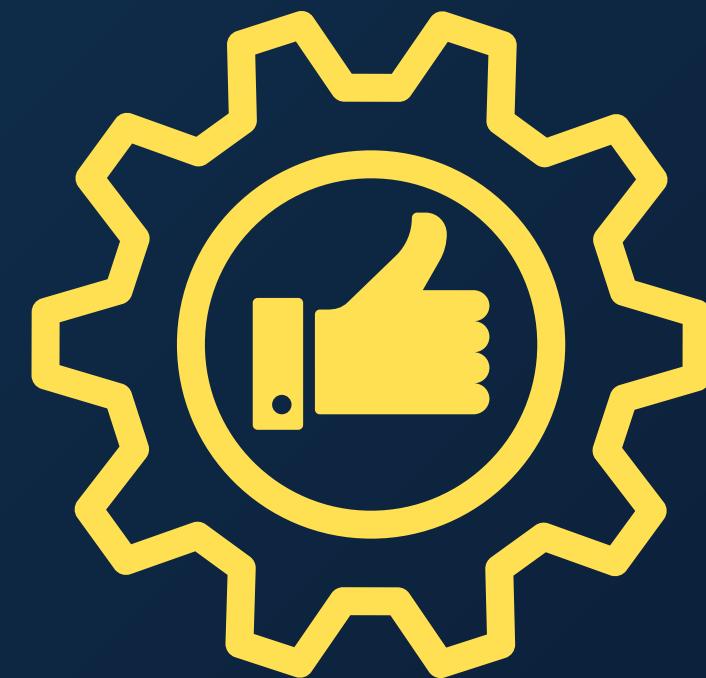
FERRAMENTAS

Bibliotecas como NumPy, SciPy e Matplotlib são utilizadas em cálculos e simulações.

CASE

A Siemens aplica algoritmos de machine learning desenvolvidos em Python para prever falhas e otimizar a manutenção de equipamentos, reduzindo custos e aumentando a eficiência das operações fabris

PRINCIPAIS VANTAGENS DO PYTHON



Simplicidade e clareza da sintaxe, facilitando o aprendizado.



Bibliotecas científicas já consolidadas, como NumPy, Pandas e Matplotlib.



Aplicações diversas, que vão da prototipagem rápida até a integração com sistemas complexos.

CONCEITOS BÁSICOS DE PROGRAMAÇÃO

Algoritmo

Sequência de passos lógicos que resolvem um problema.

Lógica computacional

Capacidade de traduzir uma ideia em instruções claras e precisas para o computador.

Linguagem de programação

Meio de comunicação entre humanos e máquinas



EXEMPLOS DO DIA A DIA

RECEITA DE BOLO (ALGORITMO CULINÁRIO)



Ingredientes → entrada de dados

Passos da Receita → processamento

Bolo pronto → saída de dados

Se a ordem ou a quantidade for alterada, o resultado muda.

EXEMPLOS DO DIA A DIA

FAZER BALIZA COM O CARRO (ALGORITMO DE DIREÇÃO)

Etapas claras

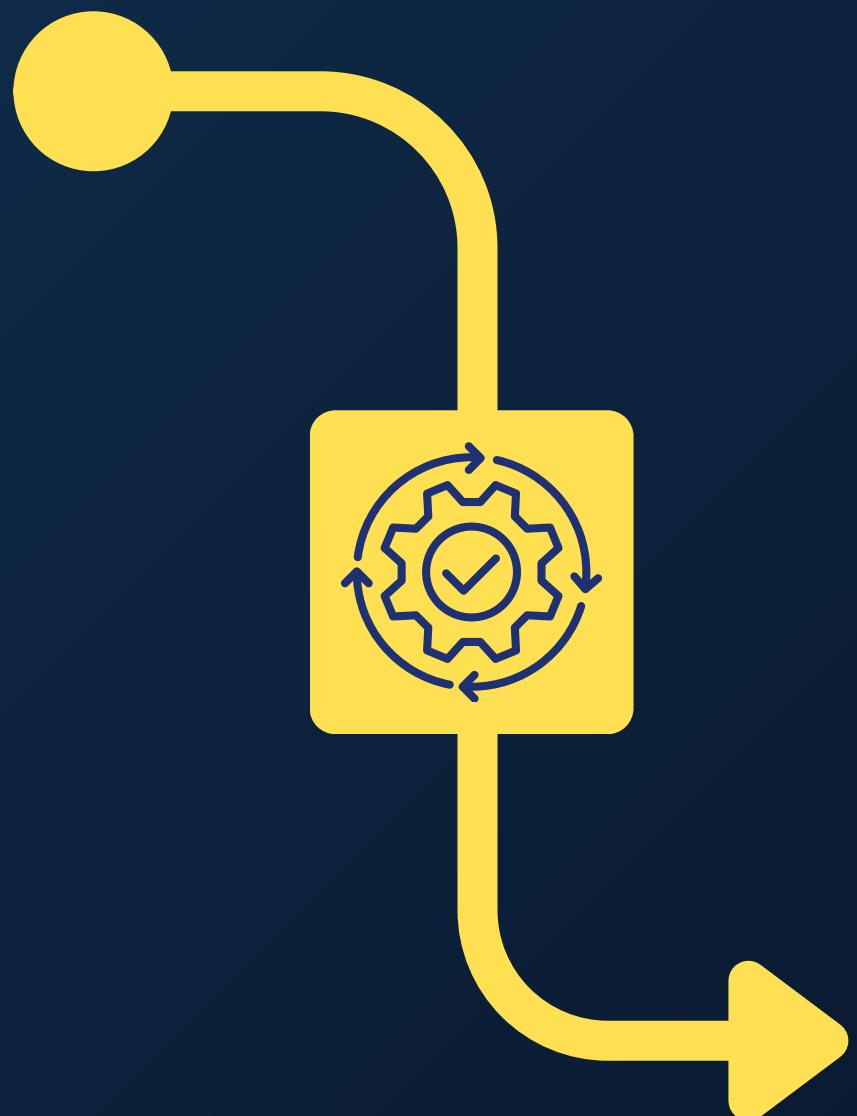
→
alinhar, girar o volante,
mover o carro, ajustar



Cada ação depende do sucesso da anterior

Resultado esperado → carro estacionado corretamente

CONCEITOS BÁSICOS DE PROGRAMAÇÃO



Em termos gerais, todo programa segue uma estrutura básica composta por três etapas: entrada, processamento e saída.

Na **entrada**, são fornecidos os dados (digitados pelo usuário, captados por sensores ou lidos de um arquivo).

No **processamento**, esses dados são manipulados de acordo com regras ou cálculos definidos no algoritmo.

Por fim, na **saída**, o programa exibe os resultados, seja em forma de números na tela, gráficos ou relatórios.

CONCEITOS BÁSICOS DE PROGRAMAÇÃO

Esses conceitos básicos são universais e independem da linguagem escolhida.

Eles formam o alicerce de toda a programação e são importantes pois permitem transformar problemas reais em soluções automatizadas, reduzindo erros, economizando tempo e ampliando a capacidade de análise.



OPERADORES ARITMÉTICOS

O Python utiliza operadores aritméticos para realizar cálculos.

Operadores aritméticos

soma (+)
subtração (-)
multiplicação (*)
divisão (/)
divisão inteira (//)
exponenciação ()**
módulo (%)

Esses operadores permitem criar programas que resolvem problemas matemáticos .

```
demo.py x
1
2 def bubble_sort(list):
3     sorted_list = list
4     is_sorted = False
5     while is_sorted == False:
6         swaps = 0
7         for i in range(len(list)):
8             if sorted_list[i] >
9                 # swap
10                temp = sorted_l
11                sorted_list[i] =
12                sorted_list[i + 1]
13                swaps += 1
14            print(swaps)
15        if swaps == 0:
16            is_sorted = True
17    return sorted_list
18
19
20 print(bubble_sort([2, 1, 3]))
21
```

INTRODUÇÃO AOS AMBIENTES DE DESENVOLVIMENTO (IDE)

Para programar em Python, precisamos de um ambiente de desenvolvimento. Existem várias opções, mas nesta disciplina utilizaremos o **Visual Studio Code (VS Code)**.

O VS Code é uma IDE leve, moderna e muito utilizada no mercado. Nele, será possível:

- Escrever código de forma organizada.
- Executar programas em Python.
- Instalar extensões que facilitam o desenvolvimento.

VARIÁVEIS E TIPOS DE DADOS

Um conceito central em programação é o de **variável**, que nada mais é do que um espaço nomeado na memória para armazenar valores. No Python, trabalharemos inicialmente com tipos de dados primitivos, como:

Inteiros (int)

Números decimais (float)

Cadeias de caracteres (str)

Valores lógicos (bool)

Compreender variáveis e tipos de dados é essencial, pois eles são usados em praticamente todos os programas que desenvolvemos.



```
"name" => null
"Surname" => null
"Username" => "admin"
"gender" => null
"email" => "info@mecanbay.com.br"
"email_verified_at" => null
"password" => "$2y$10$1rmu"
"isActive" => 1
"user_role" => "Administrador"
"avatar" => "assets/img/users/1.jpg"
"remember_token" => "0dwri"
"created_at" => "2022-01-01T14:20:00.000Z"
"updated_at" => "2022-01-01T14:20:00.000Z"
```

ENTRADA E SAÍDA DE DADOS NO PYTHON



Por fim, veremos como interagir com o usuário por meio de entradas e saídas de dados.

Para exibir informações usamos o comando `print()`.

Para receber informações digitadas pelo usuário utilizamos o comando `input()`.

Combinando esses recursos, já será possível criar programas simples que solicitam valores, realizam cálculos e retornam resultados.

Esse será o ponto de partida para resolver problemas mais complexos nas próximas aulas.

EXERCÍCIOS PRÁTICOS

OPERAÇÕES MATEMÁTICAS COM ENTRADA DE DADOS

Escreva um programa em Python que:

01

Peça ao usuário dois números inteiros

02

Calcule e exiba

- A soma dos números.
- A subtração do primeiro pelo segundo.
- A multiplicação.
- A divisão.

EXERCÍCIOS PRÁTICOS

SAUDAÇÃO PERSONALIZADA

Escreva um programa em Python que:

01

Pergunte o nome do usuário

02

Pergunte o período do dia (manhã, tarde ou noite)

03

Exiba uma saudação personalizada, como no exemplo abaixo

Digite seu nome: Ana

Qual período do dia (manhã/tarde/noite)? tarde

Olá, Ana! Tenha uma boa tarde!

ATÉ A PROXIMA

