



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

FUNÇÕES E ANÁLISE DE DADOS COM O PANDAS

POR JOÃO VÍTOR PAMPONET ESTEVES

O QUE IREMOS APRENDER

- 1 Funções em Python
- 2 Introdução à biblioteca pandas
- 3 Integração com matplotlib
- 4 Proposta do projeto final

FUNÇÕES EM PYTHON

As funções são blocos de código reutilizáveis que permitem organizar e estruturar melhor os programas.

Elas são usadas para executar tarefas específicas e podem receber valores de entrada (parâmetros) e retornar resultados.

Essa abordagem evita repetições de código e torna os programas mais claros e fáceis de manter.



def

CONCEITO DE FUNÇÃO EM PY

Uma função em Python é um bloco de código organizado que realiza uma tarefa específica.

Pense nela como uma máquina que:

- **Recebe** dados de entrada (parâmetros)
- **Processa** esses dados
- **Retorna** um resultado



ESTRUTURA PADRÃO DE UMA FUNÇÃO

```
def calcular_media(nota1, nota2, nota3):
    soma = nota1 + nota2 + nota3
    media = soma / 3
    return media

resultado = calcular_media(8, 7, 9)
print(f"Média: {resultado:.1f}")
```

Obrigatória para definir uma função
Indica ao Python: "vou criar uma função"

- **Identificador** único
Segue mesmas regras de nomes de variáveis
Use nomes **descritivos** que expliquem o que a função faz
- **Variáveis** que a função recebe
Podem ser zero, um ou vários
Separe por vírgulas

ESTRUTURA PADRÃO DE UMA FUNÇÃO

```
def calcular_media(nota1, nota2, nota3):
    soma = nota1 + nota2 + nota3
    media = soma / 3
    return media

resultado = calcular_media(8, 7, 9)
print(f"Média: {resultado:.1f}")
```

• **Obrigatório** após os parênteses
Indica o início do bloco da função

• **Bloco de código** indentado
Onde a mágica acontece
Executa a tarefa específica

(Opcional)
Devolve um resultado
Se omitido, retorna *None*
Pode retornar qualquer tipo de dado

TIPOS DE FUNÇÃO

Em Python não existe uma classificação formal ou rígida de “tipos de funções” como em outras linguagens.

A distinção é mais conceitual e baseada no comportamento da função.



def
print
return
len

FORMAS COMUNS DE CLASSIFICAÇÃO

POR RETORNO



Funções com retorno:

Usam Return para devolver um resultado

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

Funções sem retorno:

Executam ações (ex: print())-> retornam None

```
def exibir_mensagem():  
  
    print("== BEM-VINDO AO SISTEMA ==")  
    print("Operação concluída com sucesso!")  
    print("=====")
```

FORMAS COMUNS DE CLASSIFICAÇÃO

POR PARÂMETROS



Funções sem parâmetros:

```
def ligar_motor()
```

```
def saudacao_fixa():

    print("=" * 30)
    print("  BEM-VINDO AO SISTEMA!")
    print("  Tenha um excelente dia!")
    print("=" * 30)
```

Saída

```
=====
BEM-VINDO AO SISTEMA!
Tenha um excelente dia!
```

FORMAS COMUNS DE CLASSIFICAÇÃO

POR PARÂMETROS



Funções com parâmetros:
def calcular_area(largura, altura):

```
def saudacao_fixa(nome, periodo):  
  
    if periodo == "manhã":  
        mensagem = print(f"Bom dia, {nome}!")  
    elif periodo == "tarde":  
        mensagem = print(f"Boa tarde, {nome}!")  
    else:  
        mensagem = print(f"Boa noite, {nome}!")
```

FORMAS COMUNS DE CLASSIFICAÇÃO

POR ORIGEM



Funções built-in:

Já existem no Python (Ex: `print()`, `len()`)

```
print("FUNÇÕES BUILT-IN PARA TEXTO")
```

Saída

FUNÇÕES BUILT-IN PARA TEXTO

FORMAS COMUNS DE CLASSIFICAÇÃO

POR ORIGEM



Funções do usuário:

Criadas por você com def

```
def formatar_nome(nome):  
    return nome.strip().title()
```

INTRODUÇÃO À BIBLIOTECA PANDAS



O Pandas é uma das bibliotecas mais populares do Python para manipulação e análise de dados.

Ele fornece estruturas eficientes, como o DataFrame, que permite trabalhar com tabelas de dados de forma semelhante ao Excel, mas com muito mais flexibilidade.

Com o Pandas, é possível importar dados de diferentes fontes (como CSV, Excel ou bancos de dados), organizar colunas, filtrar informações, realizar cálculos estatísticos e gerar relatórios.

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

Importando o Pandas

```
import pandas as pd
```

CARREGAR UM ARQUIVO CSV

CSV

```
df = pd.read_csv('dados.csv')
```

CARREGAR UM ARQUIVO Excel

Excel (.xlsx, .xls)

```
df = pd.read_excel('planilha.xlsx')
```

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

CRIAR UM DATAFRAME

A partir de Dicionário

```
dados = {  
    'Nome': ['Ana', 'João', 'Maria', 'Pedro'],  
    'Idade': [25, 30, 22, 35],  
    'Cidade': ['SP', 'RJ', 'SP', 'BH'],  
    'Salario': [5000, 7000, 4500, 8000]  
}  
  
df = pd.DataFrame(dados)  
print(df)
```

Saída

	Nome	Idade	Cidade	Salario
0	Ana	25	SP	5000
1	João	30	RJ	7000
2	Maria	22	SP	4500
3	Pedro	35	BH	8000

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

CRIAR UM DATAFRAME

A partir de Lista de Listas

```
dados = [  
    ['Ana', 25, 'SP', 5000],  
    ['João', 30, 'RJ', 7000],  
    ['Maria', 22, 'SP', 4500]  
]
```

```
colunas = ['Nome', 'Idade', 'Cidade', 'Salario']  
df = pd.DataFrame(dados, columns=colunas)  
print(df)
```

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

CARREGAR COLUNAS DO ARQUIVO

.columns

```
print("Colunas do dataframe:")
print(df.columns)
```

Saída

```
Index(['Nome', 'Idade', 'Cidade', 'Salario'], dtype='object')
```

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

TRABALHANDO COM COLUNAS

Converter para lista

```
lista_colunas = df.columns.tolist()  
print("Lista de colunas:", lista_colunas)
```

Verificar se coluna existe

```
if 'Nome' in df.columns:  
    print("Coluna 'Nome' existe no dataframe")
```

Número de colunas

```
num_colunas = len(df.columns)  
print(f"Número de colunas: {num_colunas}")
```

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

CARREGAR LINHAS DO ARQUIVO

Primeiras Linhas

.head()

```
print("Primeiras 5 linhas:")
print(df.head())
```

Saída

	Nome	Idade	Cidade	Salario
0	Ana	25	SP	5000
1	João	30	RJ	7000
2	Maria	22	SP	4500
3	Pedro	35	BH	8000

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

CARREGAR LINHAS DO ARQUIVO

Últimas linhas

.tail()

```
print("Últimas 5 linhas:")
print(df.tail())
```

Saída

	Nome	Idade	Cidade	Salario
2	Maria	22	SP	4500
3	Pedro	35	BH	8000

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

SELECIONAR COLUNAS ESPECÍFICAS

Seleção com Colchetes

Uma coluna (retorna Series)

```
coluna_nome = df['Nome']
print("Coluna Nome:")
print(coluna_nome)
```

Saída

```
0    Ana
1    João
2   Maria
3   Pedro
Name: Nome, dtype: object
```

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

SELECIONAR COLUNAS ESPECÍFICAS

Múltiplas Colunas

Duas colunas (retorna DataFrame)

```
duas_colunas = df[['Nome', 'Idade']]  
print("Nome e Idade:")  
print(duas_colunas)
```

Saída

	Nome	Idade
0	Ana	25
1	João	30
2	Maria	22
3	Pedro	35

COMANDOS BÁSICOS DA BIBLIOTECA PANDAS

SELEÇÃO DE COLUNAS ESPECÍFICAS

Várias Colunas

Várias colunas específicas

```
colunas_selecionadas = df[['Nome', 'Cidade', 'Salario']]  
print("Colunas selecionadas:")  
print(colunas_selecionadas)
```

Saída

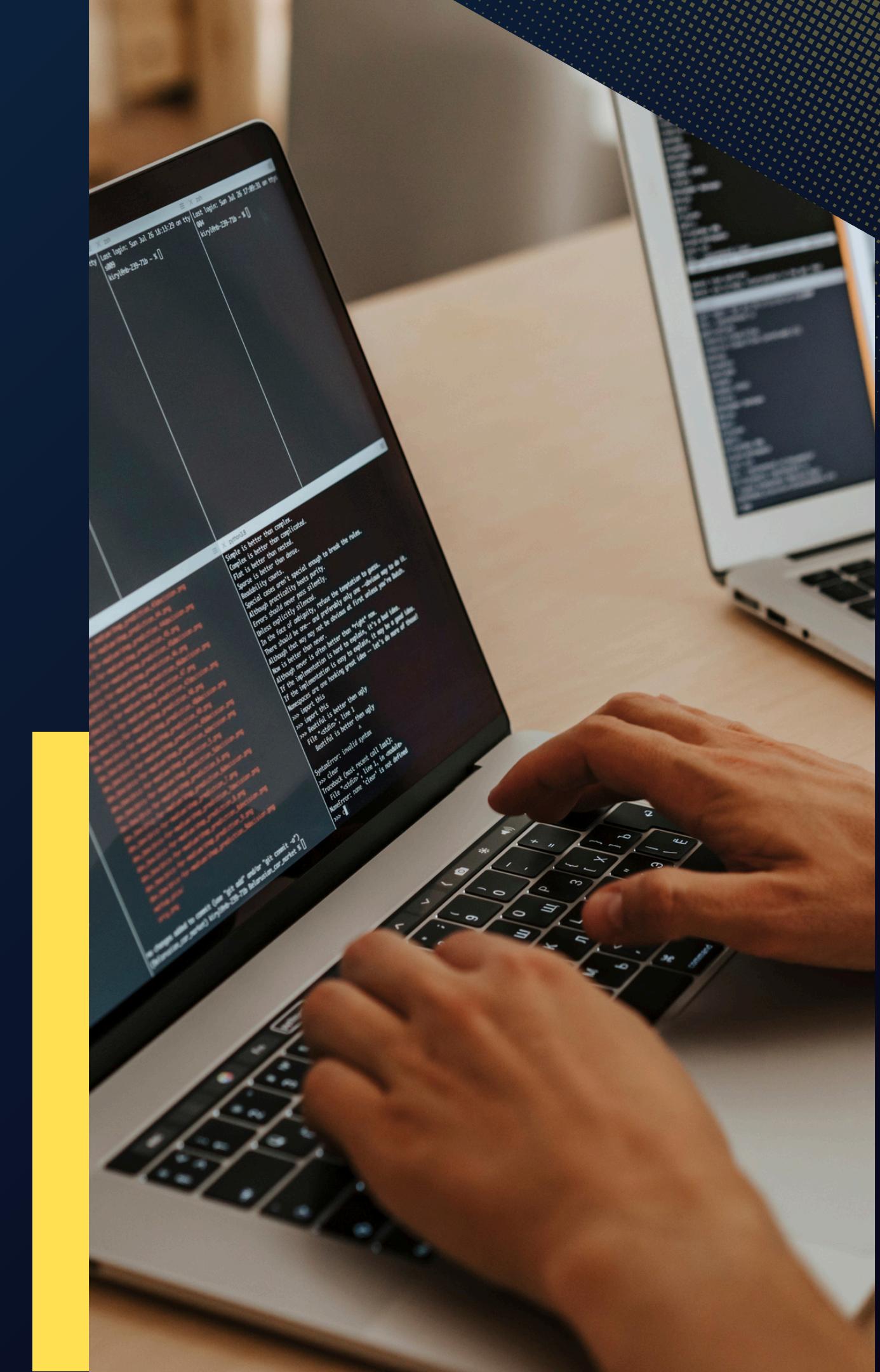
	Nome	Cidade	Salario
0	Ana	SP	5000
1	João	RJ	7000
2	Maria	SP	4500
3	Pedro	BH	8000

PROJETO FINAL

Descrição do Projeto

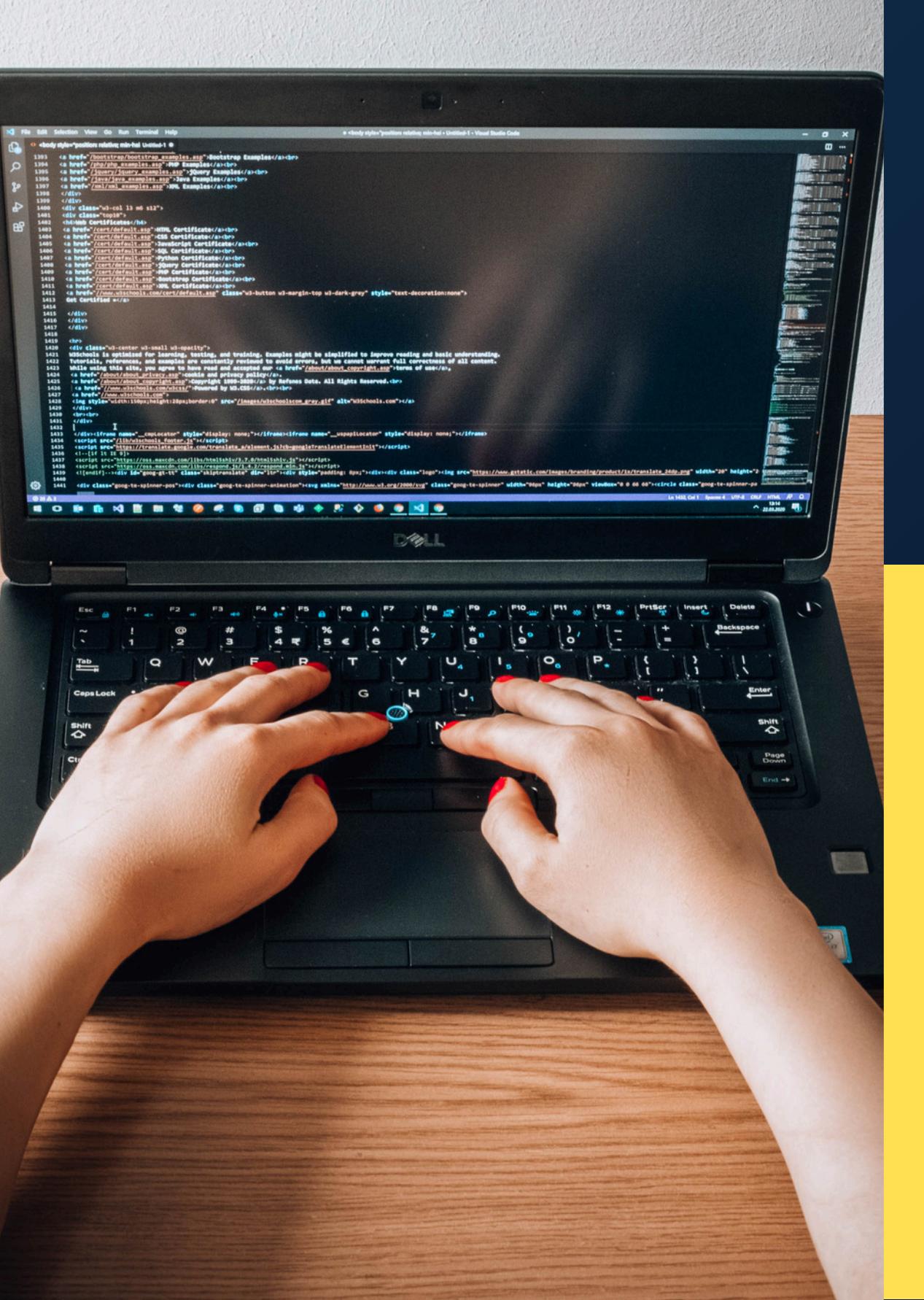
O projeto final tem como objetivo consolidar os conhecimentos adquiridos durante o curso por meio da análise de dados reais do setor industrial.

Os alunos trabalharão em equipes, utilizando o dataset Manufacturing Dataset (Kaggle) para realizar diferentes análises aplicadas à engenharia de produção, empregando Python e bibliotecas como pandas e matplotlib.



OBJETIVO GERAL

- Aplicar conceitos de manipulação e análise de dados utilizando o Python .
- Praticar leitura, filtragem, agrupamento e visualização de dados com a biblioteca pandas.
- Desenvolver habilidades de trabalho em equipe, documentação e apresentação de resultados.
- Construir um projeto prático que poderá ser incluído no portfólio profissional dos alunos.



METODOLOGIA

- A turma será dividida em equipes, sendo sorteado um tema para cada uma realizar uma análise específica com base nos dados do dataset.
- Cada equipe terá até 20 minutos para apresentar os resultados da análise, incluindo uma breve explicação da lógica utilizada no código.
- Após a apresentação, os demais alunos poderão fazer perguntas e discutir pontos relevantes do projeto.

```
8 # Use the json module to work with JSON data
9 theJSON = json.load(open('eq_data.json'))
10
11 # now we can access the contents of the JSON file
12 if "title" in theJSON["metadata"]:
13     print(theJSON["metadata"]["title"])
14
15 # output the number of events, the number of
16 # features in the JSON file
17 count = theJSON["metadata"]["count"]
18 print(str(count) + " events recorded")
19
20 # for each event, print the place where it
21 # occurred and its magnitude
22 for i in theJSON["features"]:
23     print(i["properties"]["place"])
24     print("Magnitude: " + str(i["properties"]["mag"]))
25     print("-----\n")
26
27 # print the events that only have a magnitude
28 # greater than or equal to 4.0
29 for i in theJSON["features"]:
30     if i["properties"]["mag"] >= 4.0:
31         print("Magnitude: " + str(4.1f) + " " + i["properties"]["mag"])
32         print("-----\n")
33
34 # print the events where at least 1 person
35 # reported feeling them
36 for i in theJSON["features"]:
37     if i["properties"]["felt"] > 0:
38         print("Event: " + i["properties"]["place"])
39         print("Magnitude: " + str(i["properties"]["mag"]))
40         print("Felt reports: " + str(i["properties"]["felt"]))
41         print("-----\n")
```

DATASET UTILIZADO

O **Manufacturing Dataset** é um conjunto de dados fictícios do setor industrial, reunindo informações sobre produção, qualidade, custos, desempenho operacional e condições ambientais.

Ele é ideal para estudos de análise de dados aplicados à Engenharia de Produção.

Disponível em: [Manufacturing Dataset – Kaggle](#)



TEMAS DAS ANÁLISES

Cada equipe receberá, por sorteio, um dos temas abaixo para desenvolver sua análise:

1

Produção Total e Eficiência:

Análise da quantidade de peças produzidas por produto, turno e máquina.

2

Controle de Qualidade:

investigação dos defeitos, taxa de refugo e impacto na produção.

3

Custos de Produção:

Avaliação dos custos de material, mão de obra e consumo de energia.

4

Desempenho Operacional:

Análise do tempo de produção, manutenção, períodos de parada e reprocessos.

5

Condições Ambientais:

Estudo da influência da temperatura e umidade sobre a qualidade e o volume produzido.

REQUISITOS DO PROJETO

- O código deverá ser desenvolvido em Python, utilizando obrigatoriamente as bibliotecas Pandas e Matplotlib.
- É permitida a utilização de bibliotecas adicionais, desde que a escolha seja devidamente justificada no projeto.
- A análise de dados deve incluir pelo menos uma representação visual gerada preferencialmente com a biblioteca Matplotlib.
- O código deve ser bem estruturado, comentado e legível, facilitando a compreensão por terceiros.
- Durante a apresentação, o código deverá ser executado e demonstrado em funcionamento.
- Além da apresentação, cada equipe deverá fornecer o link para um repositório público no GitHub, contendo todo o código do projeto, devidamente organizado e versionado.

AVALIAÇÃO

A nota do projeto será distribuída da seguinte forma:

Critério	Peso
Código	50%
Apresentação	30%
Documentação	20%

Observação: Apesar de o projeto ser realizado em equipe, será avaliada também a contribuição individual de cada participante.

ATÉ A PROXIMA

