Apostila do Desenvolvedor Python

Composição e Herança:

Utilize composição para reutilizar código sem criar uma hierarquia rígida.

```
class Motor:
    def ligar(self):
        print("Motor ligado")

class Carro:
    def __init__(self, motor):
        self.motor = motor

    def ligar_carro(self):
        self.motor.ligar()

motor = Motor()
carro = Carro(motor)
carro.ligar_carro()  # Saída: Motor ligado
```

Conceitos Básicos

Variáveis e Tipos de Dados:

```
nome = "João"  # String
idade = 25     # Inteiro
altura = 1.75  # Float
ativo = True   # Booleano
```

Estruturas de Controle:

```
# Condicional
if idade > 18:
    print("Maior de idade")
else:
    print("Menor de idade")

# Loop
for i in range(5):
    print(i)

while idade < 30:
    print("Ainda jovem")
    idade += 1
```

Funções:

```
def soma(a, b):
```

```
    return a + b

resultado = soma(2, 3)
print(resultado)  # Saída: 5
```

Manipulação de Arquivos:

```
# Escrever em um arquivo
with open('arquivo.txt', 'w') as f:
    f.write('Escrevendo no arquivo')

# Ler de um arquivo
with open('arquivo.txt', 'r') as f:
    conteudo = f.read()
    print(conteudo)
```

Conceitos Avançados

List Comprehensions:

```
quadrados = [x**2 for x in range(10)]
print(quadrados)  # Saída: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Decorators:

```
def meu_decorator(func):
    def wrapper():
        print("Algo antes da função")
        func()
        print("Algo depois da função")
    return wrapper

@meu_decorator
def diz_oi():
    print("Oi!")

diz_oi()
# Saída:
# Algo antes da função
# Oi!
# Algo depois da função
```

Generators:

```
def meu_generator():
    for i in range(10):
        yield i

for valor in meu_generator():
```

```
    print(valor)
```

Context Managers:

```
class MeuContexto:
    def __enter__(self):
        print("Entrando no contexto")
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        print("Saindo do contexto")

with MeuContexto() as contexto:
    print("Dentro do contexto")
# Saída:
# Entrando no contexto
# Dentro do contexto
# Saindo do contexto
```

Bibliotecas Comuns

Requests: Para realizar requisições HTTP :citation[oaicite:0]{index=0}

```
import requests
response = requests.get('https://api.github.com')
print(response.status_code)
```

Pandas: Para manipulação de dados

```
import pandas as pd
df = pd.DataFrame({'Coluna1': [1, 2], 'Coluna2': [3, 4]})
print(df)
```

NumPy: Para computação numérica

```
import numpy as np
array = np.array([1, 2, 3, 4])
print(array * 2)
```

Django/Flask: Para desenvolvimento web

```
# Flask
from flask import Flask
app = Flask(__name__)

@app.route('/')
```

```
def hello():
    return "Hello, World!"

if __name__ == '__main__':
    app.run()
```