Encapsulamento em Python:

O encapsulamento em Python refere-se à prática de esconder detalhes de implementação de uma classe e exibir apenas as partes relevantes para os usuários externos da classe. Isso é alcançado através do uso de métodos e atributos privados, protegidos e públicos.

Atributos e Métodos Privados:

Atributos e métodos privados são aqueles que começam com dois sublinhados (\_\_) antes do nome.

Eles são acessíveis apenas dentro da classe onde são definidos e não podem ser acessados diretamente fora da classe.

Atributos e Métodos Protegidos:

Atributos e métodos protegidos são aqueles que começam com um sublinhado (\_) antes do nome.

Eles podem ser acessados apenas dentro da própria classe e suas subclasses.

Atributos e Métodos Públicos:

Atributos e métodos públicos são acessíveis a partir de qualquer lugar do código.

Eles não têm prefixos especiais.

Getters e Setters:

Em Python, getters e setters são métodos utilizados para acessar e modificar atributos privados.

Eles são usados para garantir o encapsulamento, permitindo a manipulação controlada dos atributos privados.

Propriedades:

Propriedades em Python permitem que métodos sejam chamados implicitamente, como se fossem atributos.

Elas são úteis para implementar getters e setters de forma mais elegante.

Benefícios do Encapsulamento:

Melhora a segurança e robustez do código, já que evita acesso direto a partes sensíveis da classe.

Facilita a manutenção do código, permitindo que as implementações internas de uma classe sejam alteradas sem afetar o código que a utiliza.

python

class ContaBancaria:

def \_\_init\_\_(self, saldo):

self.\_\_saldo = saldo # Atributo privado

def depositar(self, valor):

self.\_\_saldo += valor

def sacar(self, valor):

if self.\_\_saldo >= valor:

self.\_\_saldo -= valor

else:

print("Saldo insuficiente.")

def get\_saldo(self): # Getter para acessar o saldo privado

return self.\_\_saldo

def set\_saldo(self, novo\_saldo): # Setter para modificar o saldo privado

self.\_\_saldo = novo\_saldo

# Exemplo de uso

conta = ContaBancaria(1000)

print(conta.get\_saldo()) # Saída: 1000

conta.depositar(500)

print(conta.get\_saldo()) # Saída: 1500

conta.sacar(2000) # Saída: Saldo insuficiente.