

# **Relatório Sistema bancário - adaptação POO**

**Aluna:** Maria Eduarda Dornelles Gobbi

## **1. Introdução**

Este relatório apresenta a adaptação e reorganização do código desenvolvido para o projeto da disciplina, cujo objetivo era implementar um sistema bancário em Python utilizando os princípios fundamentais de Programação Orientada a Objetos (POO). Foi utilizado:

- Uso de classes;
- Aplicação de abstração, encapsulamento, herança e polimorfismo;
- Separação do código em módulos (.py) distintos;
- Execução via terminal sem dependências externas.

## **2. Organização em Módulos**

O projeto foi dividido em quatro arquivos principais, cada um com responsabilidade clara:

1. **conta.py** – classe base abstrata `Conta`.
2. **conta\_corrente.py** – classe derivada `ContaCorrente`.
3. **conta\_poupanca.py** – classe derivada `ContaPoupanca`.
4. **main.py** – script principal de execução via terminal.

Essa separação melhora a manutenção, reutilização e leitura do código

## **3. Aplicação dos Conceitos de POO**

### **3.1. Abstração**

Foi criada a classe abstrata `Conta` contendo atributos comuns:

`titular, número e saldo.`

E métodos genéricos como `depositar()` e `sacar()`.

Essa classe não representa diretamente uma conta real, funcionando apenas como modelo – caracterizando abstração.

### **3.2. Encapsulamento**

Os atributos foram protegidos com *underscore* (ex.: `_saldo`), e o acesso foi controlado por métodos.

Isso impede modificações indevidas diretamente pelo usuário e deixa o código mais seguro.

### 3.3. Herança

As classes:

- `ContaCorrente`
- `ContaPoupanca`

herdam de **Conta**, reaproveitando atributos e comportamentos e evitando duplicação de código.

### 3.4. Polimorfismo

O método `sacar()` foi sobreescrito nas subclasses, permitindo comportamentos diferentes:

- **Conta Corrente**: permite saque com limite especial.
- **Conta Poupança**: só permite saque se houver saldo suficiente.

O mesmo método é chamado no menu, mas cada classe responde de acordo com sua própria regra — ou seja, polimorfismo.

## 4. Interação com o Usuário via Terminal

O arquivo `main.py` contém um menu simples que permite ao usuário:

- Criar conta corrente ou poupança;
- Depositar;
- Sacar;
- Ver saldo;
- Listar contas criadas.

As operações são feitas através de um loop e leitura de inputs, sem bibliotecas externas, atendendo às regras do projeto.

## 5. Conclusão

O código foi completamente modificado da forma exigida, usando os conceitos de POO, e executando corretamente no terminal.

