

PROJETO FINAL

Documento de Descrição e Motivação

Disciplina: Programação Orientada a Objetos (Python), SENAI Goiás

Alunos: [Integrante 1] e [Integrante 2]

Projeto: Controle de Assinaturas

1. O problema

Hoje é fácil perder a conta de quantos serviços a gente assina (streaming, academia, aplicativos, revistas) e de quanto isso pesa no bolso todo mês. Muita gente só percebe o total quando bate o olho na fatura do cartão. O Controle de Assinaturas é um aplicativo de desktop que reúne todas as assinaturas de uma pessoa num só lugar e mostra, na hora, quanto ela gasta por mês somando tudo.

2. Motivação

Escolhemos esse tema porque é um problema real que nós dois vivemos: perder o controle dos gastos recorrentes. É simples de entender, mas rende uma modelagem interessante: nem toda assinatura é cobrada do mesmo jeito. Umas são mensais e outras anuais, então o app precisa saber converter tudo para um custo mensal comparável antes de somar. Isso encaixa naturalmente com o que aprendemos: uma hierarquia de tipos de assinatura (POO), um banco para guardar os dados (SQLite) e uma janela para cadastrar e visualizar (Tkinter).

3. Funcionalidades planejadas

- Cadastrar uma assinatura (nome, valor e tipo: mensal ou anual).
- Listar todas as assinaturas cadastradas.
- Editar e remover uma assinatura.
- Mostrar o custo mensal total (somando tudo, com as anuais divididas por 12).
- Manter os dados salvos entre uma execução e outra (banco de dados).

4. Como o projeto usa o que aprendemos

Herança e polimorfismo: a classe base Assinatura e duas subclasses, AssinaturaMensal e AssinaturaAnual. As duas têm o método `custo_mensal()`, mas cada uma calcula do seu jeito. Para somar o total, o app percorre a lista e chama `custo_mensal()` em cada assinatura sem perguntar o tipo dela.

Exemplo: uma Netflix mensal de R\$ 39,90 devolve 39,90; um Spotify anual de R\$ 480,00 devolve 40,00 (480 dividido por 12). O total soma as duas: R\$ 79,90.

Abstração: não existe “assinatura genérica”. Assinatura é uma classe abstrata (ABC) e declara `custo_mensal()` como método abstrato, então o Python não deixa criar um objeto Assinatura direto e obriga toda subclasse a implementar o cálculo.

Exemplo: se amanhã criarmos `AssinaturaSemestral` e esquecermos o `custo_mensal()`, o programa acusa o erro na hora, em vez de somar errado.

Encapsulamento: o valor não é alterado direto. Ele passa por `property/setter`, que recusa número negativo levantando `ValueError`.

Exemplo: tentar `assinatura.valor = -10` dá erro em vez de gravar um valor inválido.

Banco de dados (SQLite): uma tabela assinaturas guarda `id`, `nome`, `valor` e `tipo`. O app cadastra, lista, edita e remove.

Exemplo: a linha `(1, 'Netflix', 39.90, 'mensal')` representa a assinatura da Netflix no banco.

Interface (Tkinter): uma janela com os campos `Nome` e `Valor`, um seletor de tipo (`mensal/anual`), o botão `Adicionar` e a lista das assinaturas cadastradas.

Exemplo: depois de cadastrar as duas, a janela mostra em destaque “Custo mensal total: R\$ 79,90”.