UNIVERSIDADE DA REGIÃO DE JOINVILLE - UNIVILLE

Alunos: João Breis, Samuel Leal, Aramis Otero, Kleber Souza e Marcelo Andrejov

Paradigmas de Programação

1. Sistema Escolhido

O sistema escolhido foi um gerenciador simples de tarefas. Ele permite cadastrar, listar e concluir tarefas.

2. Paradigma 1: Programação Orientada a Objetos (POO)

A linguagem utilizada foi Python. O paradigma orientado a objetos é baseado na criação de classes e objetos, o que facilita a modelagem de entidades do mundo real.

Código (resumo):

```
class Tarefa:
    def __init__(self, titulo, concluida=False):
        self.titulo = titulo
        self.concluida = concluida

    def concluir(self):
        self.concluida = True

class GerenciadorTarefas:
    def __init__(self):
        self.tarefas = []
```

```
    def adicionar(self, tarefa):
        self.tarefas.append(tarefa)

    def listar(self):
        return [(t.titulo, t.concluida) for t in self.tarefas]
```

A escolha deste paradigma se justifica pela clareza na estruturação do código, facilidade de manutenção e aderência ao modelo de entidades com estados e comportamentos.

3. Paradigma 2: Programação Funcional

A linguagem utilizada foi JavaScript com abordagem funcional. Este paradigma prioriza funções puras e evita a mutabilidade de dados.

Código (resumo):

```
const criarTarefa = (titulo) => ({ titulo, concluida: false });

const concluirTarefa = (tarefa) => ({ ...tarefa, concluida: true });

const adicionarTarefa = (lista, tarefa) => [...lista, tarefa];

const listarTarefas = (lista) =>
  lista.map((t) => `${t.titulo} - ${t.concluida ? 'Concluída' : 'Pendente'}`);
```

A opção por esse paradigma se deu pela ênfase em funções puras, o que reduz efeitos colaterais e torna o código mais fácil de testar e manter.

4. Comparação entre as Abordagens

- Organização: POO é estruturada em classes; Funcional é baseada em funções.

- Estado: POO usa atributos mutáveis; Funcional evita mutabilidade.

- Reusabilidade: POO utiliza herança e composição; Funcional reutiliza funções.

- Curva de aprendizado: POO é mais intuitiva para iniciantes; Funcional exige mudança de lógica.

5. Dificuldades Encontradas

- POO: Criar hierarquias e manter coesão entre objetos pode tornar o sistema mais complexo.

- Funcional: A lógica funcional pura demanda atenção ao lidar com listas e estruturas imutáveis, o que pode aumentar a verbosidade.

6. Vantagens e Desvantagens

POO:

- Pontos positivos: Organização clara, fácil de entender e modificar quando o sistema cresce.

- Pontos negativos: Pode acabar ficando muito complicado para sistemas simples, com muita coisa pra manter.

Funcional:

- Pontos positivos: Código mais limpo, fácil de testar e com menos chances de erro.

- Pontos negativos: Às vezes fica difícil de entender, principalmente pra quem está começando, e pode parecer meio confuso.