Programação Orientada a Objetos (POO)

**FECAP**

**Caio Gomes 24026876**  
**Vitor Rodrigues dos Santos 23025502**

São Paulo

2025

## Introdução

A entrega se concentra na criação das classes principais, interações entre elas, e implementação de métodos como getters, setters, e outros conceitos fundamentais da POO.

## Objetivo da Implementação

A principal tarefa foi desenvolver as classes responsáveis pela integração e orquestração do sistema. As classes foram projetadas para garantir o fluxo adequado das operações no sistema e assegurar a modularidade e reutilização de código, características essenciais da POO.

### Funcionalidades Implementadas

1. **Criação de Classes**: Foram criadas as classes principais do sistema, representando os objetos e entidades que o sistema manipula.
2. **Encapsulamento**: Os dados das classes foram encapsulados, utilizando atributos privados e métodos públicos (getters e setters) para garantir o controle de acesso e modificação desses dados.
3. **Herança**: Foram implementadas classes-filhas que herdam atributos e métodos de classes-pai, permitindo a reutilização de código e a criação de funcionalidades específicas.
4. **Polimorfismo**: Foi utilizado o conceito de polimorfismo para permitir que métodos com o mesmo nome se comportem de maneira diferente, dependendo do contexto.
5. **Construtores**: As classes foram implementadas com construtores para garantir a inicialização correta dos objetos.

## Explicação do Código

### Classe Principal

using System;

public class Evento

{

// Atributos privados

private string nome;

private string data;

private string local;

// Construtor

public Evento(string nome, string data, string local)

{

this.nome = nome;

this.data = data;

this.local = local;

}

// Getters e Setters

public string GetNome()

{

return nome;

}

public void SetNome(string nome)

{

this.nome = nome;

}

public string GetData()

{

return data;

}

public void SetData(string data)

{

this.data = data;

}

public string GetLocal()

{

return local;

}

public void SetLocal(string local)

{

this.local = local;

}

// Método para exibir informações do evento

public void ExibirInfo()

{

Console.WriteLine($"Evento: {nome}");

Console.WriteLine($"Data: {data}");

Console.WriteLine($"Local: {local}");

}

}

class Program

{

static void Main(string[] args)

{

// Criando um objeto Evento

Evento evento1 = new Evento("Show de Música", "2025-03-20", "Arena da Cidade");

// Exibindo informações do evento

evento1.ExibirInfo();

evento1.SetNome("Festival de Música");

evento1.SetData("2025-04-10");

evento1.SetLocal("Estádio Nacional");

}

}

### Explicação do Código

* **Classe Evento**: A classe Evento é um exemplo de uma classe simples que representa um evento com atributos como nome, data, e local. Ela contém métodos para acessar e modificar esses atributos, conhecidos como getters e setters.
* **Construtor**: O construtor é utilizado para inicializar os objetos da classe com valores específicos na criação.
* **Encapsulamento**: Atributos privados (private) e métodos públicos para acessar e modificar esses atributos garantem que o código seja seguro e controlado.