

Fundamentos de Desenvolvimento com C#

Professor: Orlando Fonseca



Kaike Torres da silva

17.03.2025

Análise e Desenvolvimento de Sistemas (ADS)

INTRODUÇÃO

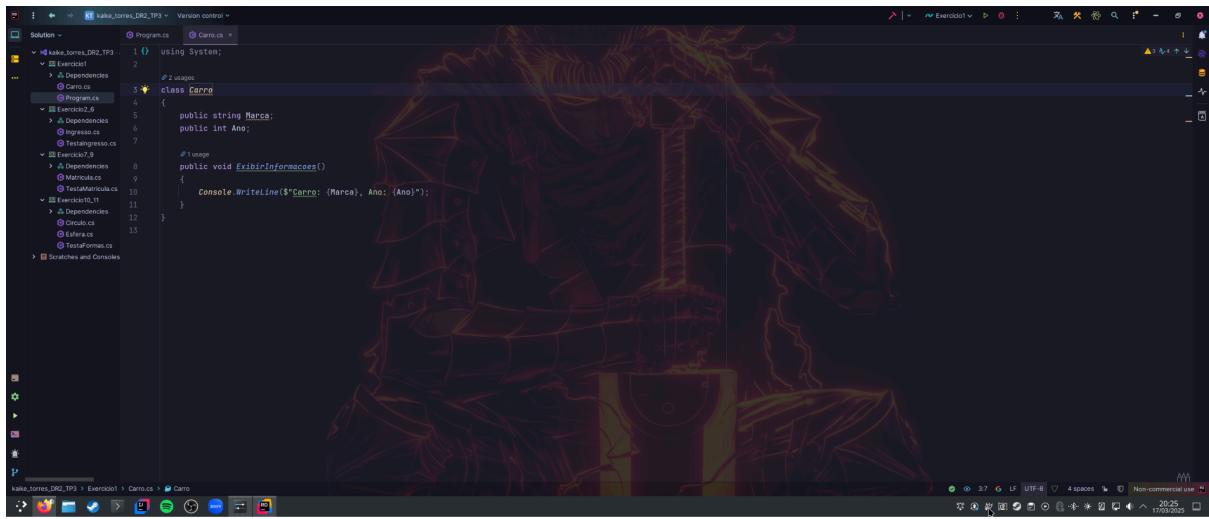
Este conjunto de exercícios tem como objetivo introduzir conceitos fundamentais da programação orientada a objetos (POO) em C#. Através da criação e manipulação de classes, atributos e métodos, os exercícios abordam desde a estrutura básica de uma classe até a implementação de cálculos matemáticos em objetos.

Exercício 1 - Conceitos de Classe, Objeto, Campos e Métodos (C#)

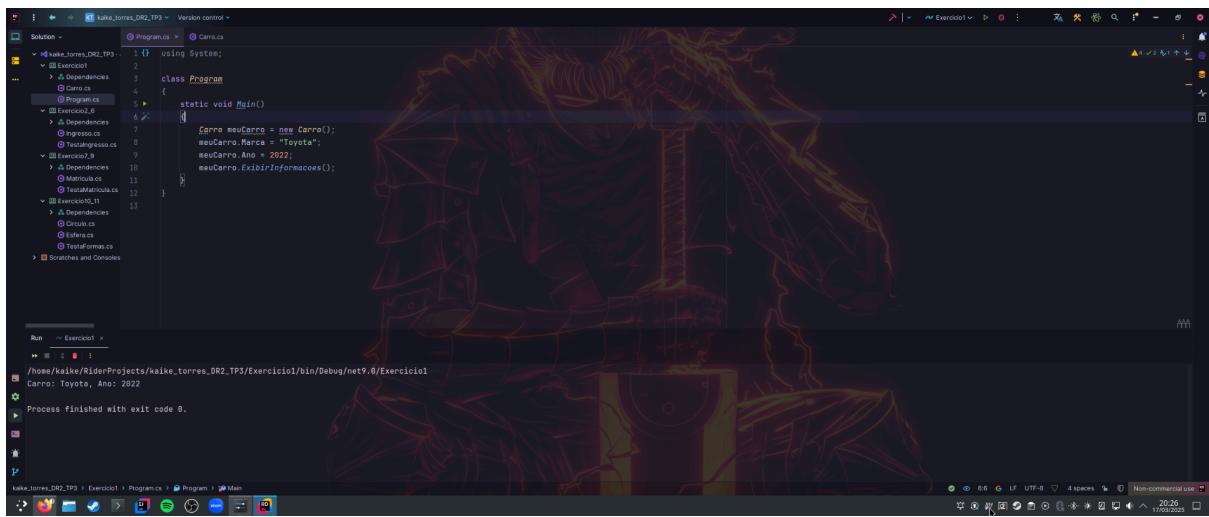
Pacote: Exercicio1

A classe **Carro** é a principal estrutura deste código e contém os atributos **Marca** e **Ano**, representando as características de um carro. A classe também possui o método **ExibirInformacoes()**, que imprime os detalhes do veículo no console.

A classe **MainProgram** contém o método **Main**, onde um objeto da classe **Carro** é instanciado com os valores "Toyota" para **Marca** e 2022 para **Ano**. O método **ExibirInformacoes()** é então chamado para exibir os dados do carro.



```
1  using System;
2
3  class Carro
4  {
5      public string Marca;
6      public int Ano;
7
8      public void ExibirInformacoes()
9      {
10         Console.WriteLine($"Carro: {Marca}, Ano: {Ano}");
11     }
12 }
```



```
1  using System;
2
3  class Program
4  {
5      static void Main()
6      {
7          Carro meuCarro = new Carro();
8          meuCarro.Marca = "Toyota";
9          meuCarro.Anو = 2022;
10         meuCarro.ExibirInformacoes();
11     }
12 }
```

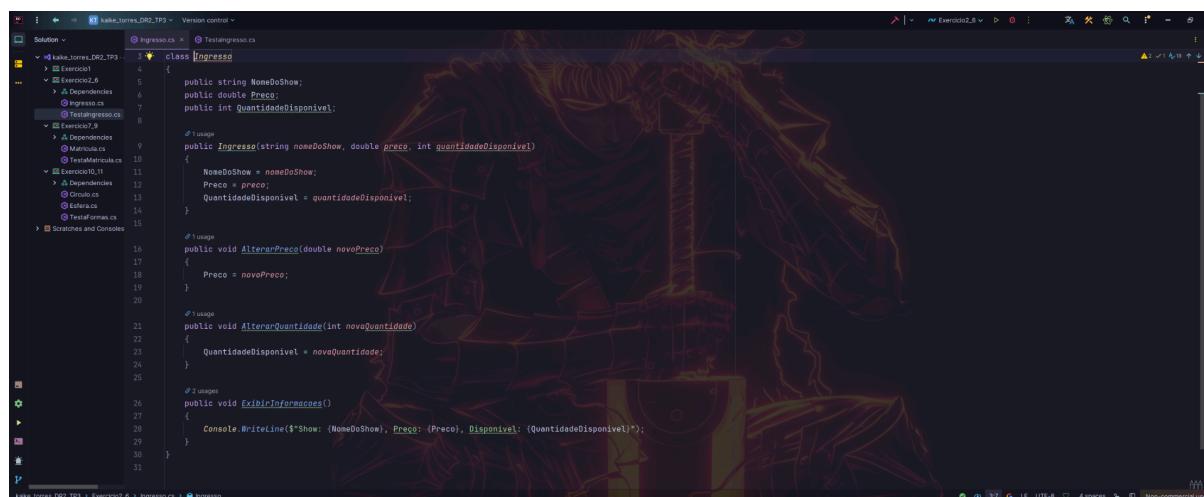
```
Carro: Toyota, Ano: 2022
Process finished with exit code 0.
```

Exercício 2 - Criando a Classe "Ingresso"

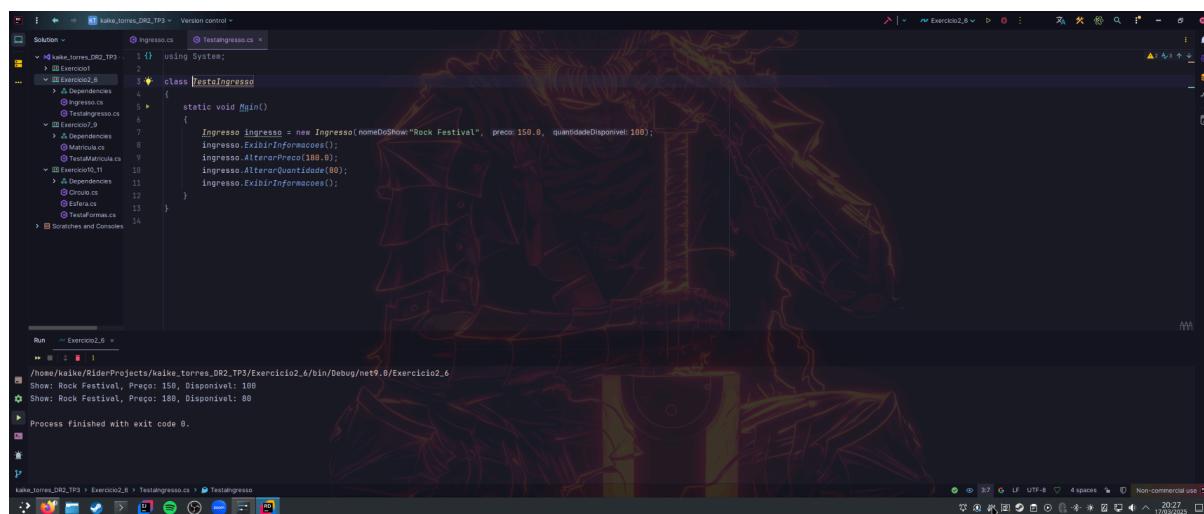
Pacote: Exercicio2_6

A classe `Ingresso` representa um ingresso de evento, contendo os atributos `NomeDoShow`, `Preco` e `QuantidadeDisponivel`. Além disso, a classe tem métodos para alterar o preço e a quantidade de ingressos disponíveis.

A classe `TestaIngresso` contém o método `Main`, onde um objeto da classe `Ingresso` é criado com os valores "Rock Festival", 150.0 e 100 para `NomeDoShow`, `Preco` e `QuantidadeDisponivel`, respectivamente. O método `ExibirInformacoes()` é chamado antes e depois das alterações no preço e na quantidade.



```
1 // Ingresso.cs
2 class Ingresso
3 {
4     public string NomeDoShow;
5     public double Preco;
6     public int QuantidadeDisponivel;
7
8     //<usage>
9     public Ingresso(string nomeDoShow, double preco, int quantidadeDisponivel)
10    {
11        NomeDoShow = nomeDoShow;
12        Preco = preco;
13        QuantidadeDisponivel = quantidadeDisponivel;
14    }
15
16    //<usage>
17    public void AlterarPreco(double novoPreco)
18    {
19        Preco = novoPreco;
20    }
21
22    //<usage>
23    public void AlterarQuantidade(int novaQuantidade)
24    {
25        QuantidadeDisponivel = novaQuantidade;
26    }
27
28    //<usage>
29    public void ExibirInformacoes()
30    {
31        Console.WriteLine($"Show: {NomeDoShow}, Preco: {Preco}, Disponivel: {QuantidadeDisponivel}");
32    }
33 }
```



```
1 // TestaIngresso.cs
2 using System;
3
4 class TestaIngresso
5 {
6     static void Main()
7     {
8         Ingresso ingresso = new Ingresso(nomeDoShow:"Rock Festival", preco: 150.0, quantidadeDisponivel: 100);
9
10        ingresso.ExibirInformacoes();
11        ingresso.AlterarPreco(180.0);
12        ingresso.AlterarQuantidade(80);
13        ingresso.ExibirInformacoes();
14    }
15 }
```

```
Run -> Exercicio2_6
Process started with exit code 0.

/kaka_torres_DR2_TP3 > TestaIngresso.cs > TestaIngresso
Show: Rock Festival, Preco: 150, Disponivel: 100
Show: Rock Festival, Preco: 180, Disponivel: 80
Process finished with exit code 0.

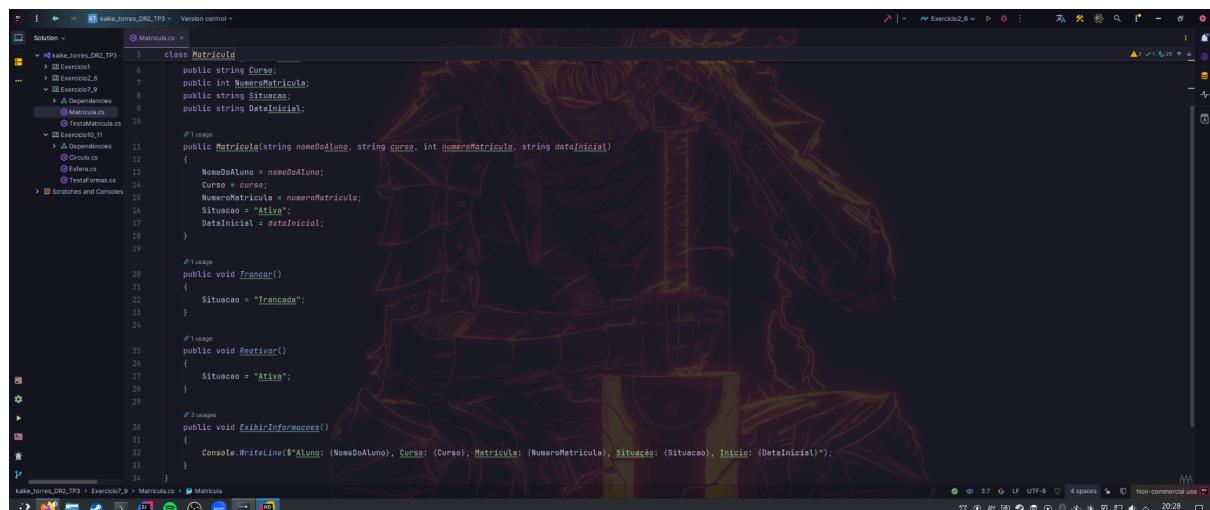
/kaka_torres_DR2_TP3 > Exercicio2_6 > TestaIngresso.cs > TestaIngresso
```

Exercício 7 - Modelando uma Matrícula

Pacote: Exercicio7_9

A classe `Matricula` modela o registro de um aluno em um curso. Ela contém os atributos `NomeDoAluno`, `Curso`, `NumeroMatricula`, `Situacao` e `DataInicial`. Além disso, possui métodos para alterar a situação da matrícula (`Trancar()` e `Reativar()`).

A classe `TestaMatricula` contém o método `Main`, onde um objeto da classe `Matricula` é criado com os valores "Kaike Torres", "ADS", 12345 e "01/03/2025". O método `ExibirInformacoes()` é chamado para exibir a situação inicial e após as mudanças de status da matrícula.



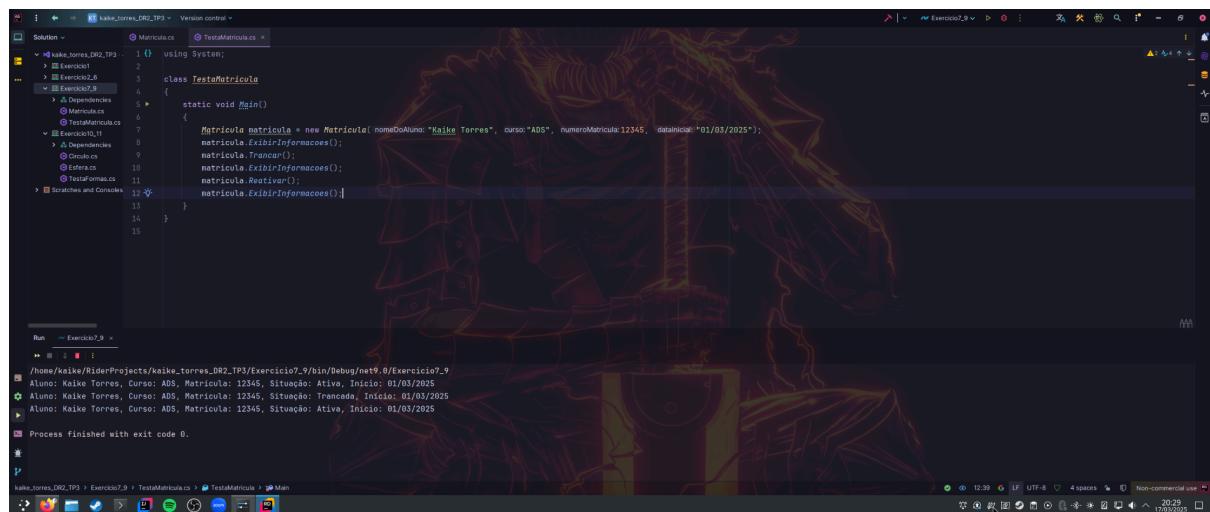
```
class Matricula
{
    public string Curso;
    public int NumeroMatricula;
    public string Situacao;
    public string DataInicial;

    //1 usage
    public Matricula(string nomeDoAluno, string curso, int numeroMatricula, string dataInicial)
    {
        NomeDoAluno = nomeDoAluno;
        Curso = curso;
        NumeroMatricula = numeroMatricula;
        Situacao = "Ativa";
        DataInicial = dataInicial;
    }

    //1 usage
    public void Trancar()
    {
        Situacao = "Trancada";
    }

    //1 usage
    public void Reativar()
    {
        Situacao = "Ativa";
    }

    //0 usages
    public void ExibirInformacoes()
    {
        Console.WriteLine($"Aluno: {NomeDoAluno}, Curso: {Curso}, Matrícula: {NumeroMatricula}, Situação: {Situacao}, Início: {DataInicial}");
    }
}
```



```
class TestaMatricula
{
    static void Main()
    {
        Matricula matricula = new Matricula(nomeDoAluno: "Kaike Torres", curso: "ADS", numeroMatricula: 12345, dataInicial: "01/03/2025");
        matricula.ExibirInformacoes();
        matricula.Trancar();
        matricula.ExibirInformacoes();
        matricula.Reativar();
        matricula.ExibirInformacoes();
    }
}
```

Output window:

```
Aluno: Kaike Torres, Curso: ADS, Matrícula: 12345, Situação: Ativa, Início: 01/03/2025
Aluno: Kaike Torres, Curso: ADS, Matrícula: 12345, Situação: Trancada, Início: 01/03/2025
Aluno: Kaike Torres, Curso: ADS, Matrícula: 12345, Situação: Ativa, Início: 01/03/2025
```

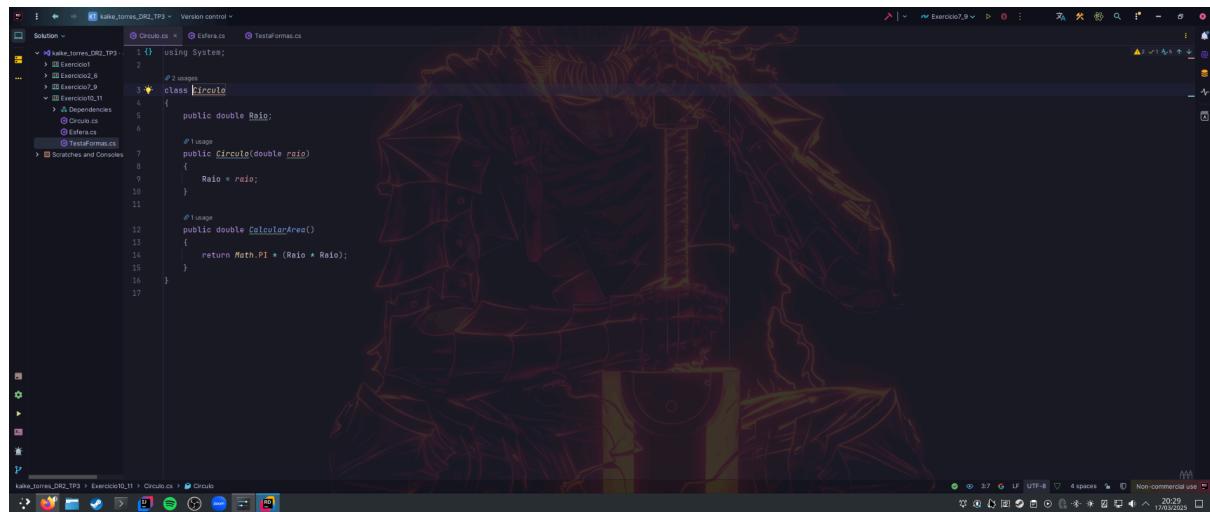
Exercício 10 - Definindo Classes de Formas Geométricas

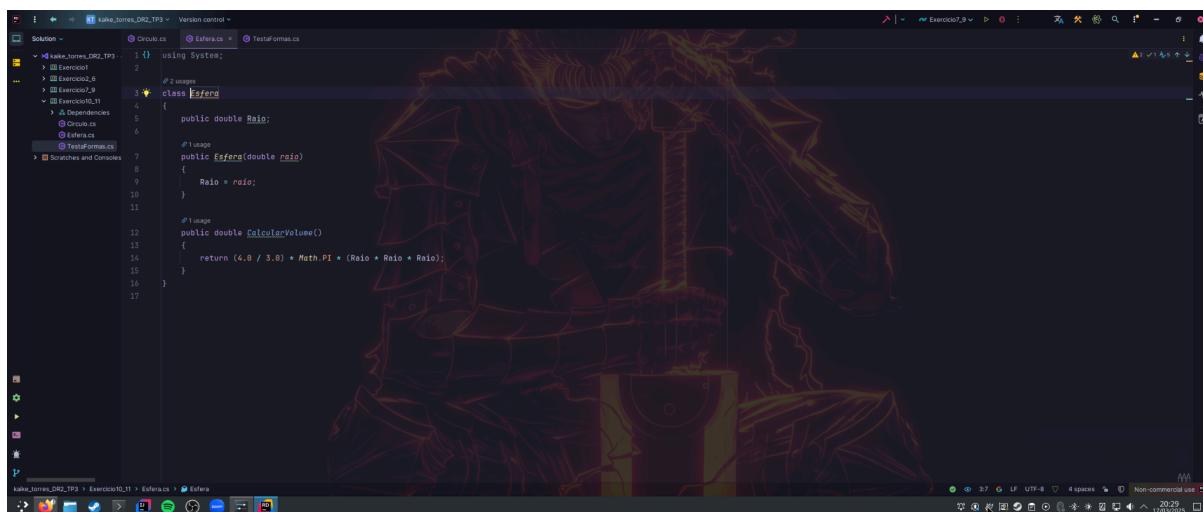
Pacote: Exercicio10_11

A classe `Circulo` representa um círculo e contém o atributo `Raio`, que é fundamental para cálculos geométricos. Ela possui o método `CalcularArea()`, que retorna a área do círculo usando a fórmula `Math.PI * (Raio * Raio)`.

A classe `Esfera` modela uma esfera e também possui o atributo `Raio`. Ela contém o método `CalcularVolume()`, que retorna o volume da esfera utilizando a fórmula `(4.0 / 3.0) * Math.PI * (Raio * Raio * Raio)`.

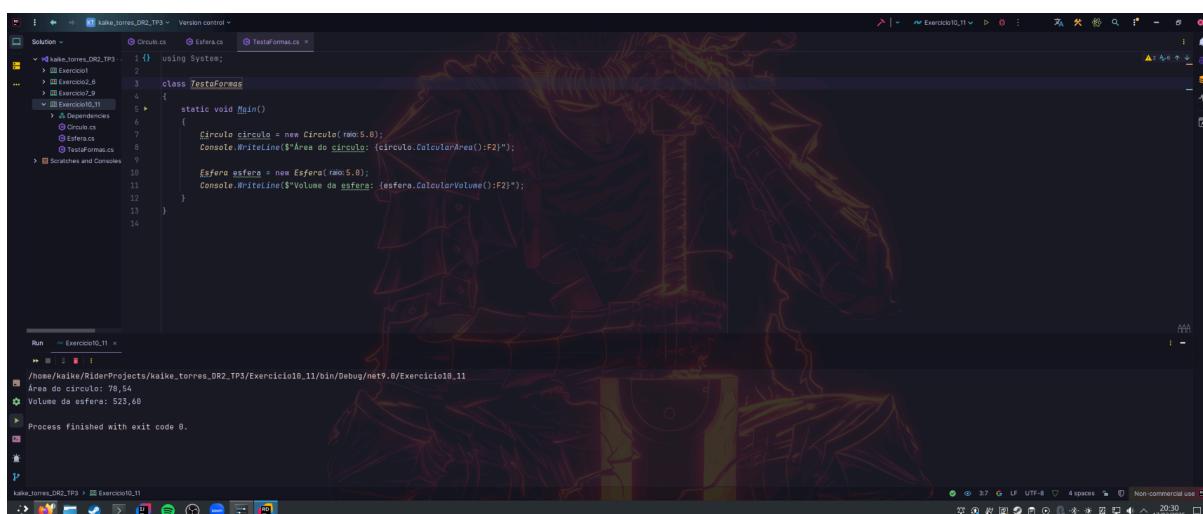
A classe `TestaFormas` contém o método `Main`, onde objetos das classes `Circulo` e `Esfera` são criados e utilizados para calcular e exibir a área e o volume, respectivamente.





A screenshot of the Visual Studio IDE showing the code for the `Esfera.cs` class. The code defines a class `Esfera` with a constructor that takes a radius and initializes it, and a method `CalcularVolume` that returns the volume of the sphere.

```
1 // Esfera.cs
2
3 class Esfera
4 {
5     public double Raio;
6
7     //1 usage
8     public Esfera(double raio)
9     {
10         Raio = raio;
11     }
12
13     //1 usage
14     public double CalcularVolume()
15     {
16         return (4.0 / 3.0) * Math.PI * (Raio * Raio * Raio);
17     }
18 }
```



A screenshot of the Visual Studio IDE showing the `TesteFormas.cs` test class. It contains a `Main` method that creates a circle with a radius of 5.0 and prints its area, then creates a sphere with a radius of 5.0 and prints its volume.

```
1 // TesteFormas.cs
2
3 class TesteFormas
4 {
5     static void Main()
6     {
7         Circulo circulo = new Circulo(Raio5.0);
8         Console.WriteLine("Área do círculo: {0:F2}", circulo.CacularArea());
9
10        Esfera esfera = new Esfera(Raio5.0);
11        Console.WriteLine("Volume da esfera: {0:F2}", esfera.CacularVolume());
12    }
13 }
14
```

The output window shows the results of the program execution:

```
Área do círculo: 78,54
Volume da esfera: 523,60
```

Conclusão

Os exercícios apresentados permitiram um aprendizado prático sobre a programação orientada a objetos em C#. Desde a criação de classes simples até a implementação de cálculos matemáticos, foi possível compreender como organizar e estruturar código de maneira modular e reutilizável.