# TESTE DE PERFORMANCE 1


**C#**

Johanna Liza Herrera

# Exercício 1: Entender a Relação entre C# e .NET

Objetivo: Compreender como C# funciona dentro do ecossistema .NET.

1. **Pesquise e escreva um resumo explicando:**
   - O que é .NET?

     .NET é um conjunto de ferramentas criado pela Microsoft para facilitar a criação de programas e aplicativos. Ele permite desenvolver diferentes tipos de softwares, como sites, aplicativos para celular e programas de computador. Além disso, funciona em diferentes sistemas operacionais, como Windows, macOS e Linux.

   - Como o C# se integra ao .NET?

     C# é uma linguagem de programação usada para escrever os códigos que dão vida aos programas dentro do .NET. Quando um programador escreve algo em C#, esse código não é entendido diretamente pelo computador. Em vez disso, ele passa por um processo que o transforma em uma linguagem intermediária, que depois é executada pelo .NET. Isso torna o código mais seguro e compatível com outros programas feitos na mesma plataforma.

   - O papel do CLR (Common Language Runtime) e do FCL

O CLR (Common Language Runtime) funciona como um tradutor e organizador do código que foi escrito pelo programador. Ele garante que o código seja compreendido pelo computador e cuida de tarefas importantes, como liberar memória automaticamente e evitar falhas no programa.

O FCL (Framework Class Library) é como uma grande caixa de ferramentas cheia de funções prontas para serem usadas. Em vez de criar tudo do zero, os programadores podem aproveitar essas ferramentas para fazer o programa funcionar mais rápido e de maneira mais eficiente. Isso inclui desde comandos básicos, como ler e salvar arquivos, até funcionalidades mais avançadas, como conectar o programa à internet ou trabalhar com gráficos.

## Exercício 2: Componentes Necessários para o Desenvolvimento Web com C#

Objetivo: Identificar os componentes que são necessários para iniciar o desenvolvimento web em C#.

1. **Faça uma lista dos seguintes componentes e descreva a função de cada um no desenvolvimento web:**

- ASP.NET Core

  ASP.NET Core é um framework para criar aplicações web modernas em C#, incluindo APIs, sites dinâmicos e aplicativos full-stack. Ele é multiplataforma (Windows, Linux e macOS) e oferece suporte a segurança, autenticação, roteamento e injeção de dependências.

- Entity Framework Core

  Entity Framework Core é um ORM que simplifica a interação entre C# e bancos de dados, permitindo operações CRUD sem escrever SQL manualmente. Suporta bancos como SQL Server, MySQL e PostgreSQL.

- Razor Pages

  Razor Pages é um modelo do ASP.NET Core que simplifica a criação de páginas dinâmicas, combinando C# e HTML para desenvolver sites interativos sem a complexidade do MVC.

## Exercício 3: Diferenças entre IDEs para Desenvolvimento em C#

Objetivo: Explorar e comparar diferentes IDEs disponíveis para desenvolvimento em C#.

1. **Compare as características do Visual Studio, Visual Studio Code e Rider.**

   *O Visual Studio é uma IDE robusta, ideal para grandes projetos em C# e .NET, oferecendo uma ampla gama de ferramentas. No entanto, pode ser mais pesada e difícil de usar para iniciantes.

   *O Visual Studio Code é leve, rápido e altamente configurável, sendo perfeito para projetos menores ou para quem prefere um ambiente de desenvolvimento flexível. Porém, exige configurações extras para trabalhar de forma completa com C#.

   *O Rider é uma IDE poderosa voltada para .NET e C#, com ótimas funcionalidades de autocompletar código e integração com Unity. Contudo, é uma ferramenta paga e tende a ser mais pesada que o Visual Studio Code.

2. **Liste vantagens e desvantagens de cada IDE considerando:**
    - Suporte a múltiplas linguagens
    - Integração com ferramentas de desenvolvimento
    - Facilidade de uso para iniciantes

   **Visual Studio**

*Ventajas:

Suporte completo ao .NET e C#

Ferramentas avançadas de depuração e design

Edição gratuita (Community) com muitos recursos

*Desventajas

Pode ser pesado e consumir muitos recursos

**Visual Studio Code**

*Ventajas:

Leve, rápido e multiplataforma

Extensível com plugins para C# e .NET

Bom para projetos menores e desenvolvimento geral

*Desventajas:

Precisa de extensões para um ambiente completo de C#

**Rider (JetBrains)**

*Ventajas:

Excelente desempenho e integração com .NET

Inteligência de código avançada e produtividade alta

Ótima opção para desenvolvimento com Unit

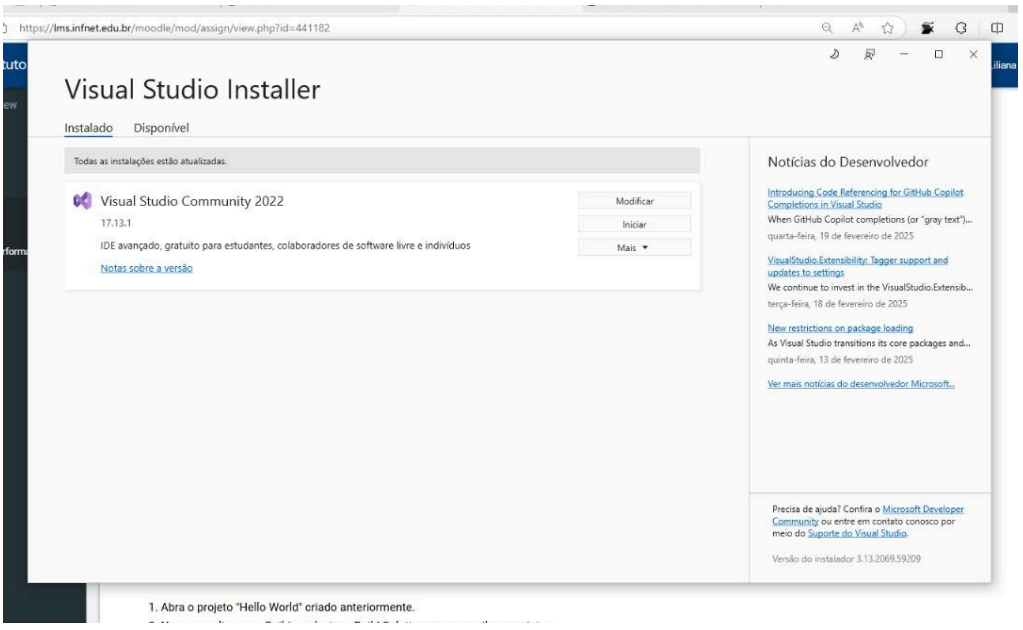
*Desventajas:

Pago, o que pode ser um obstáculo para alguns usuários

## Exercício 4: Instalar o Visual Studio Community 2022

Objetivo: Instalar Visual Studio Community 2022 com configurações para desenvolvimento em C#.

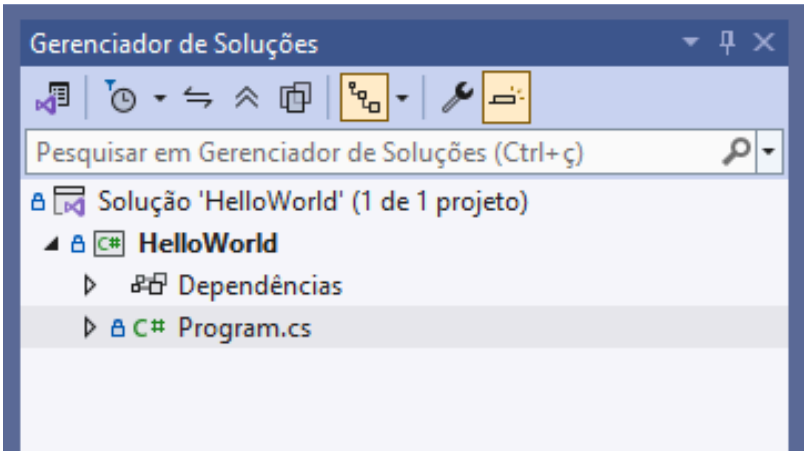
1. Baixe e instale o Visual Studio Community 2022.
2. Durante a instalação, selecione "Desktop development with C#" para instalar os componentes necessários.
3. Verifique a instalação abrindo o Visual Studio e criando um novo projeto de console.

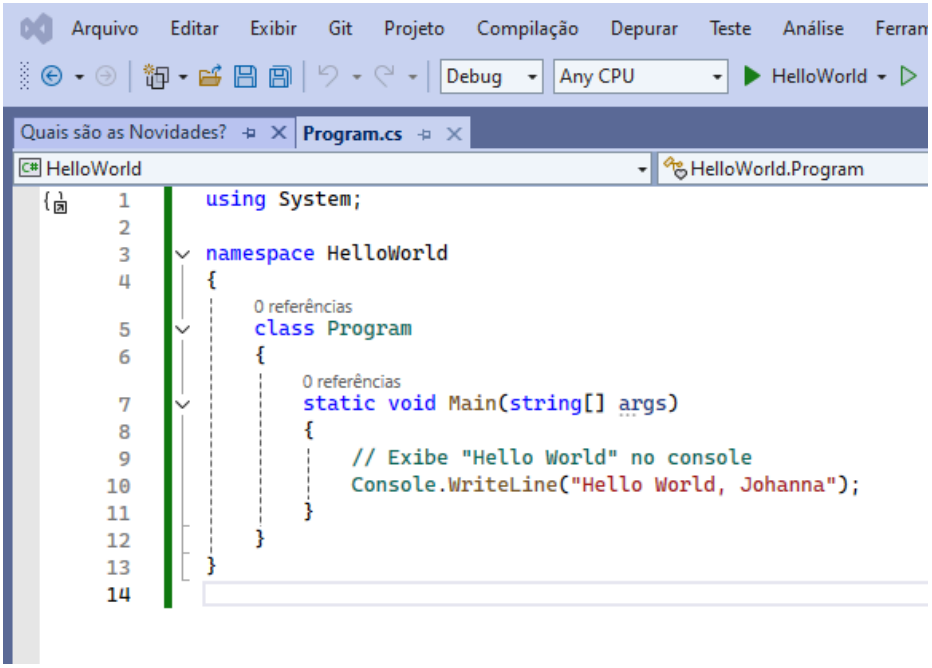


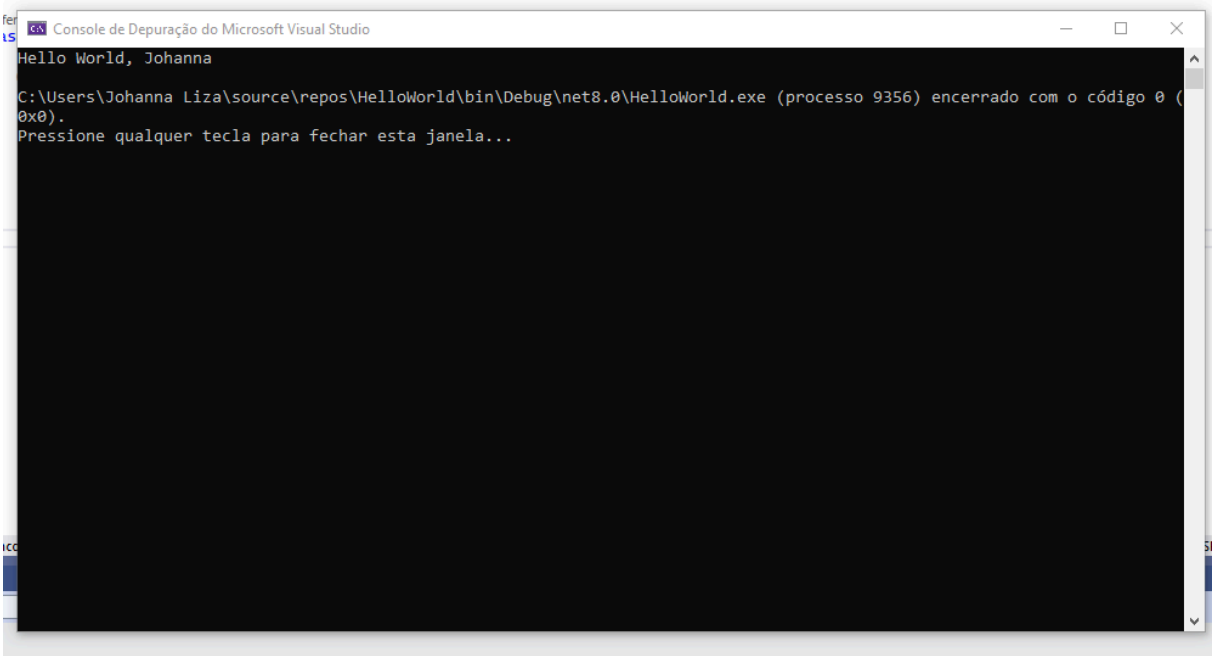

## Exercício 5: Criar um Programa "Hello World"

Objetivo: Criar um programa básico em C# que exibe "Hello World".

1. Inicie um novo projeto de Console App em C# no Visual Studio.
2. No método Main, adicione Console.WriteLine("Hello World");.
3. Compile e execute o programa para verificar a saída no console.



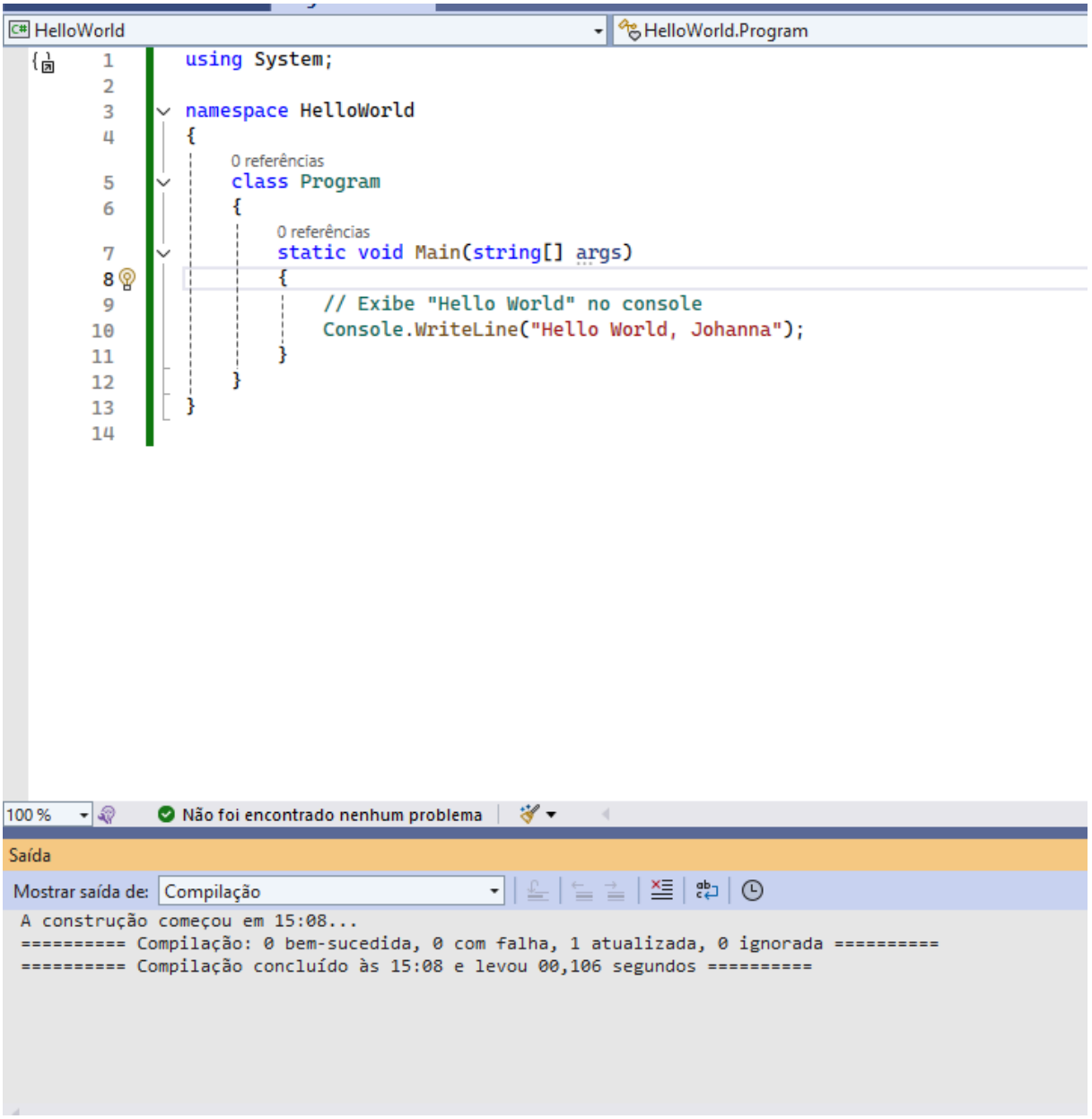



# Exercício 6: Compilar um Código Usando Visual Studio

Objetivo: Praticar a compilação de um programa usando o Visual Studio.

1. Abra o projeto "Hello World" criado anteriormente.
2. No menu, clique em Build e selecione Build Solution para compilar o projeto.
3. Confira no "Output" se não existem erros de compilação.

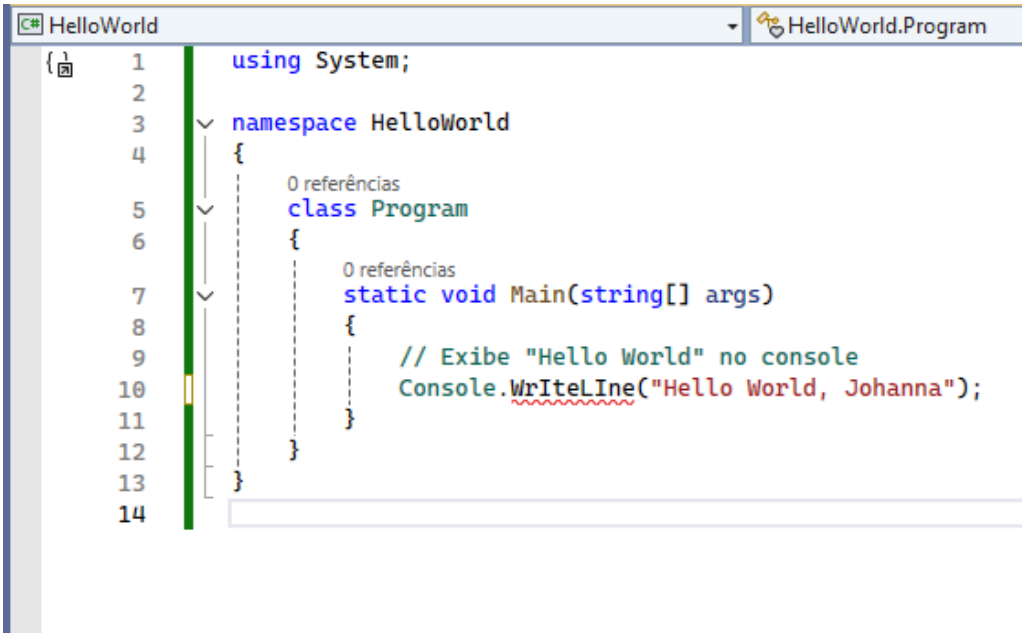



## Exercício 7: Depurar um Programa Simples

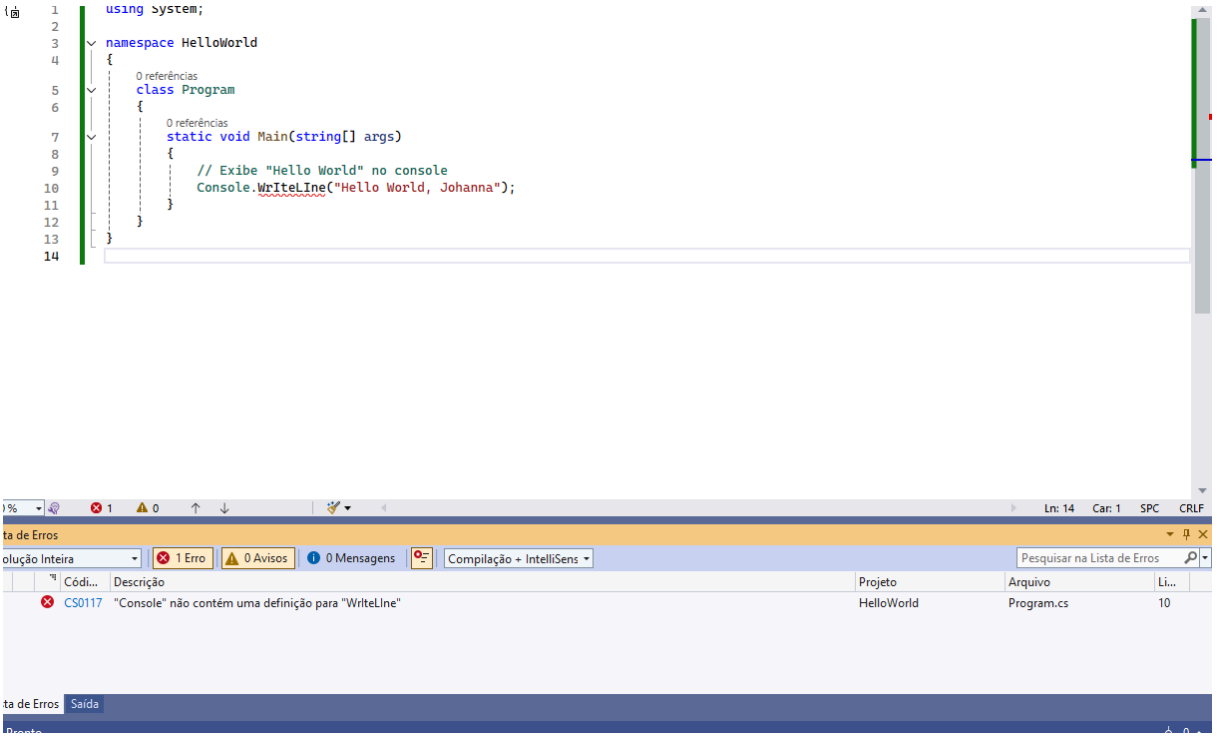
Objetivo: Usar o Visual Studio para depurar um programa que contém erros.

1. Introduza um erro intencional, como Console.Writeline ("Hello, world!");

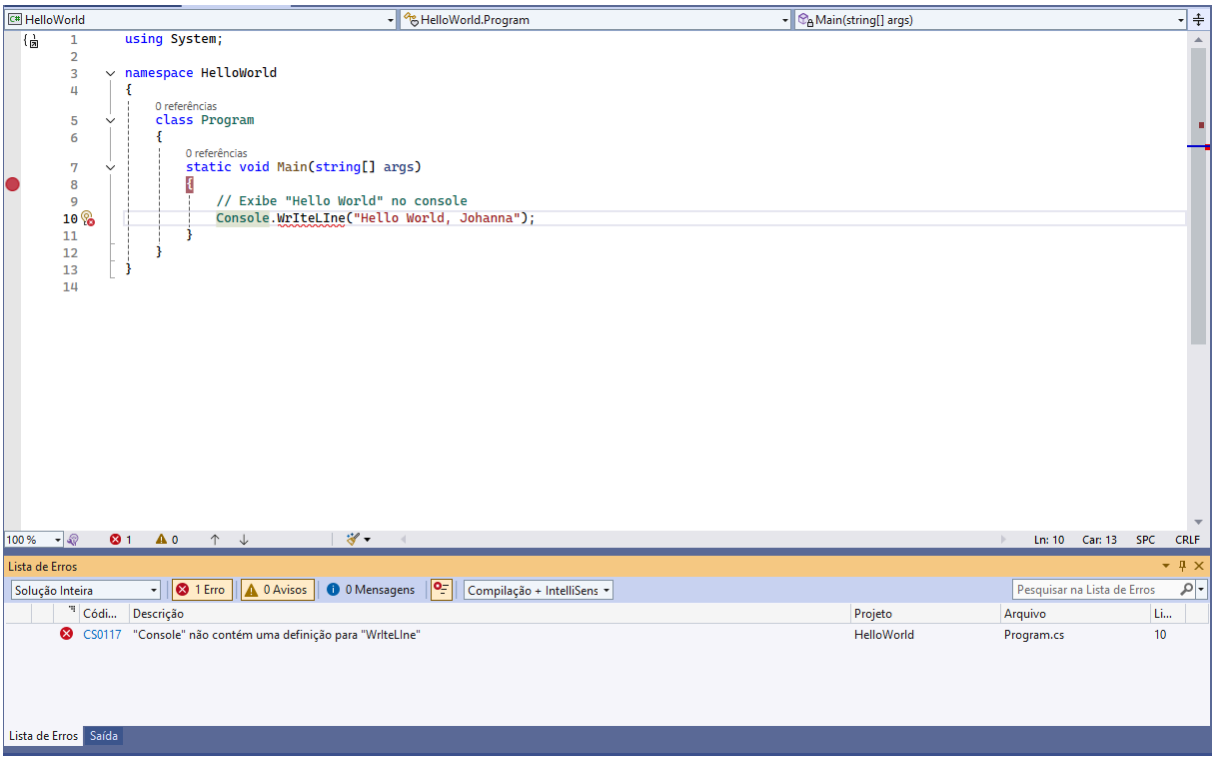
   (Writeline com 'l' maiúsculo).




2. Tente compilar e anote os erros mostrados.



3. Corrija o erro usando a ferramenta de depuração (debugger) colocando um breakpoint e acompanhando a execução.

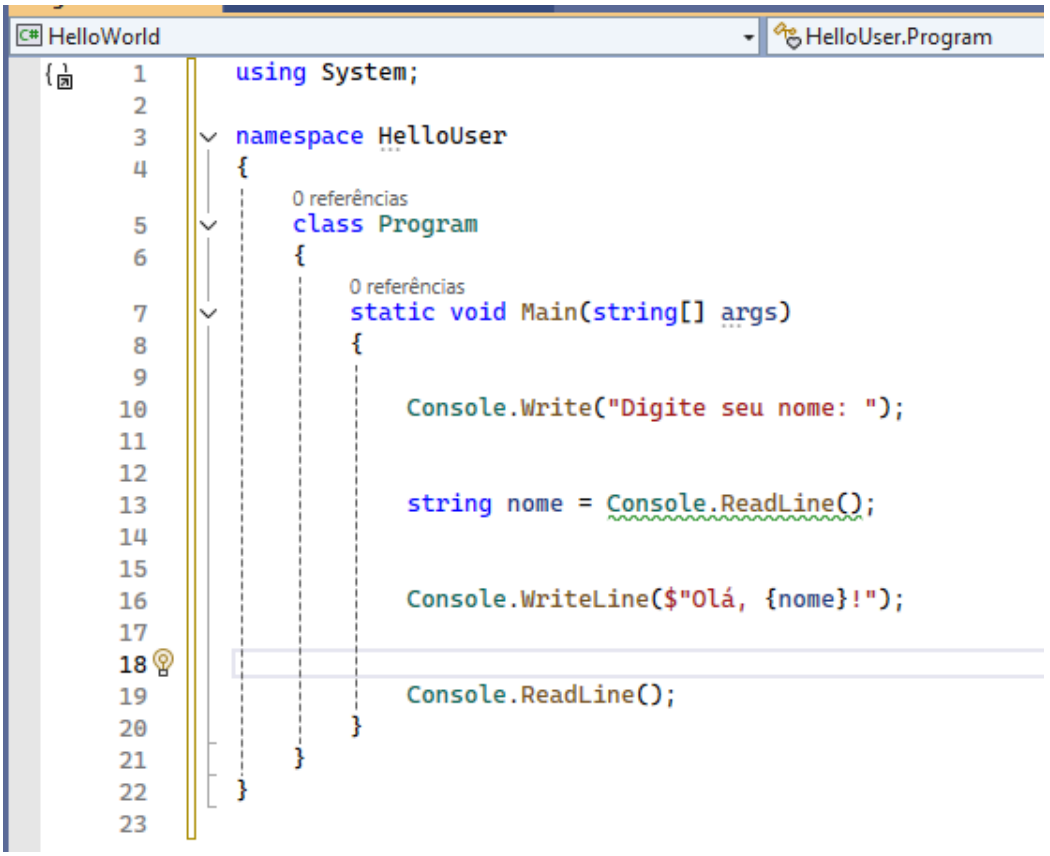


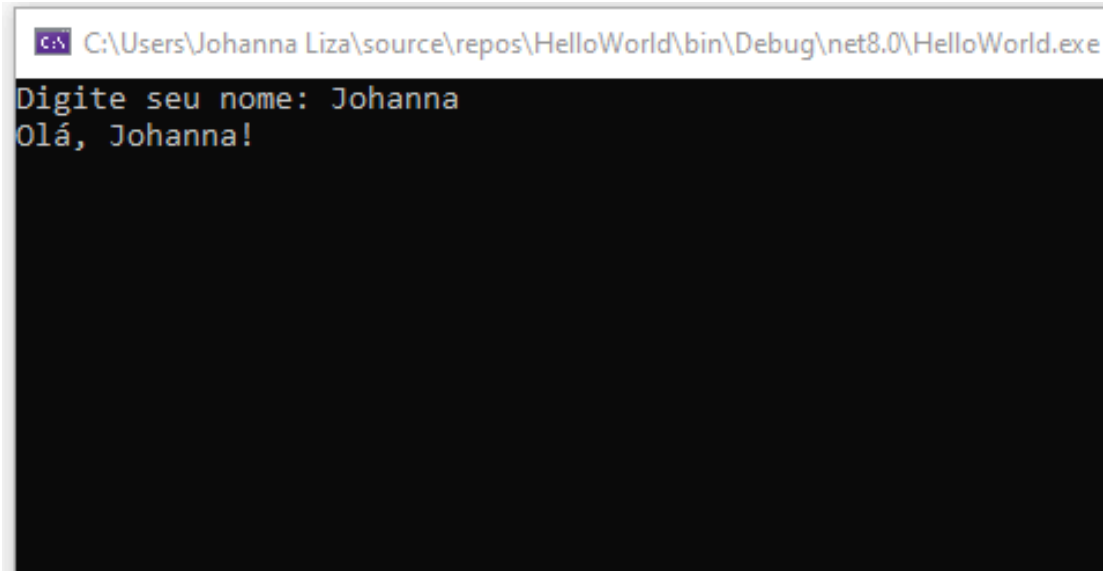

## Exercício 8: Ler Entrada do Usuário

Objetivo: Criar um programa que lê a entrada do usuário.

1. Modifique o programa "Hello World" para pedir o nome do usuário.
2. Utilize Console.ReadLine() para capturar a entrada do usuário.
3. Imprima "Olá, [nome]!" substituindo [nome] pela entrada do usuário.

4. Compile e execute para testar a funcionalidade.

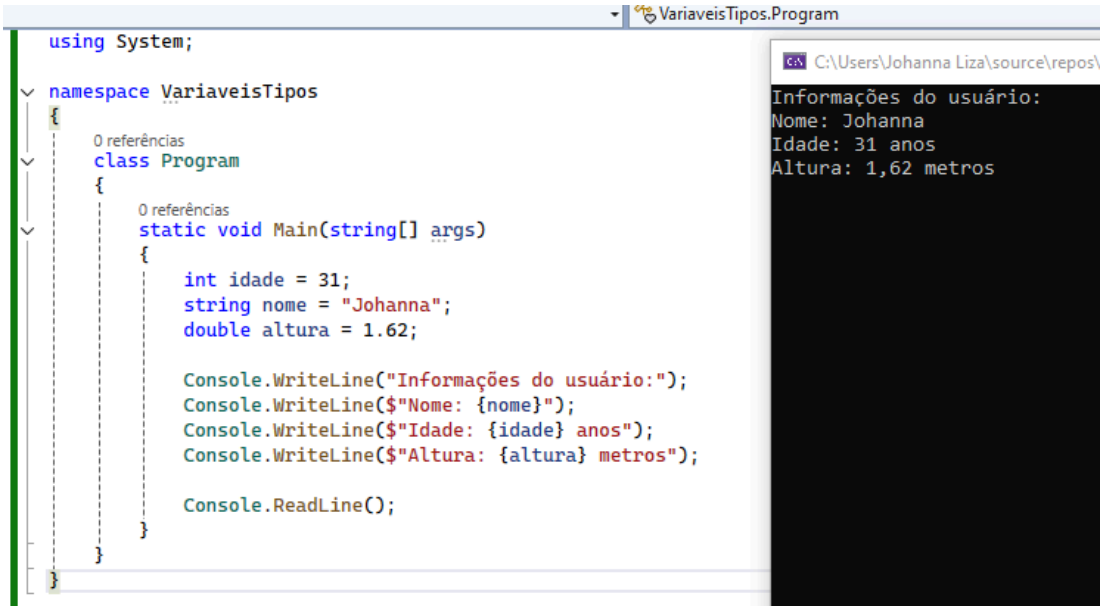



## Exercício 9: Uso de Variáveis e Tipos

Objetivo: Praticar declaração de variáveis e entender tipos de dados em C#.

1. Crie um novo programa que:
    - Declara uma variável int para idade, uma string para nome, e uma double para altura.
    - Atribua valores a estas variáveis e imprima-as usando Console.WriteLine.
2. Compile e execute o programa para verificar se os valores são impressos corretamente.



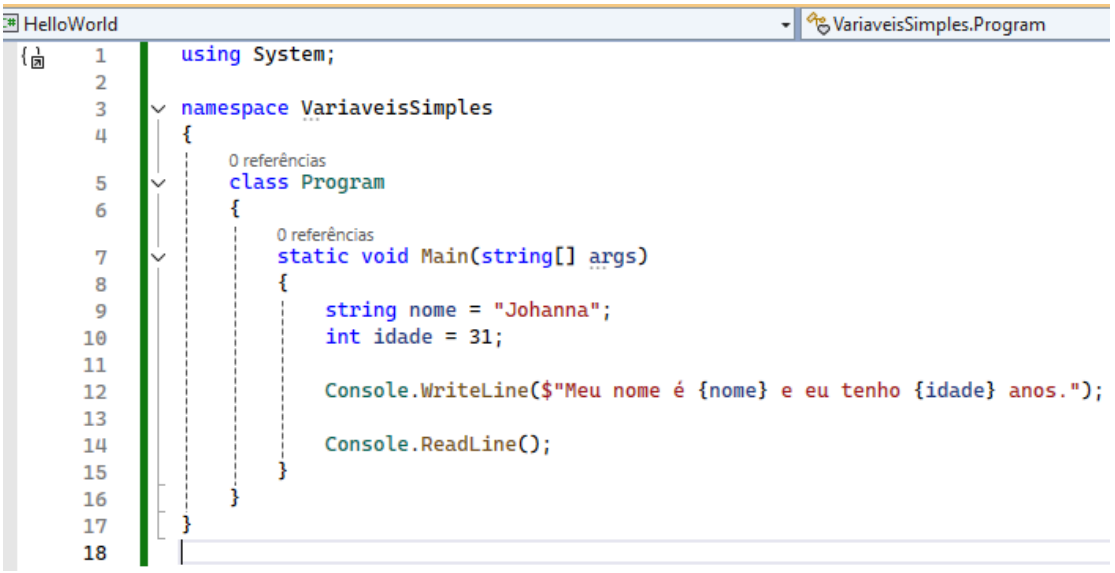

## Exercício 10: Uso de Variáveis e Tipos Simples

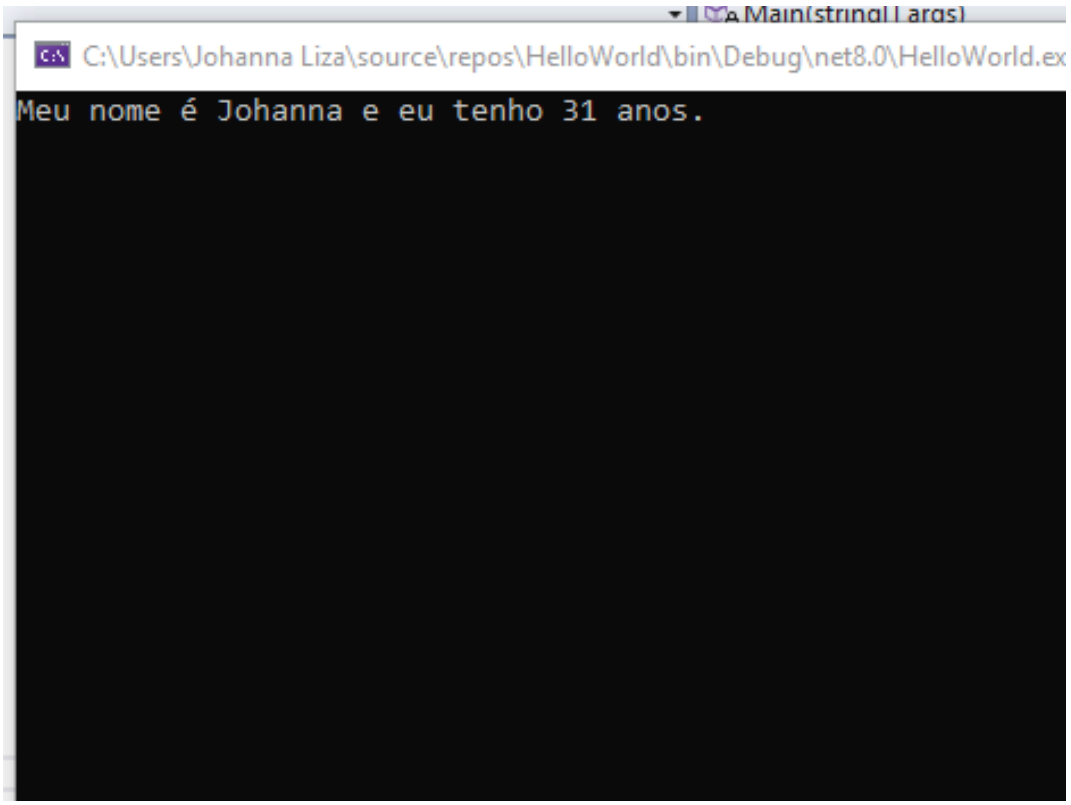
Objetivo: Praticar a declaração de variáveis e impressão de seus valores.

1. Crie um novo programa em C# que:

- Declara duas variáveis: uma string para armazenar um nome e umintpara armazenar uma idade.
- Atribua valores a estas variáveis (o nome pode ser seu próprio nome e a idade pode ser qualquer número).
- Use Console.WriteLine para imprimir uma mensagem que inclua esses valores, como "Meu nome é [nome] e eu tenho [idade] anos".

2. Compile e execute o programa para verificar a correta impressão das variáveis.




Main(string[] args)
C:\Users\Johanna Liza\source\repos\HelloWorld\bin\Debug\net8.0\HelloWorld.ex
Meu nome é Johanna e eu tenho 31 anos.