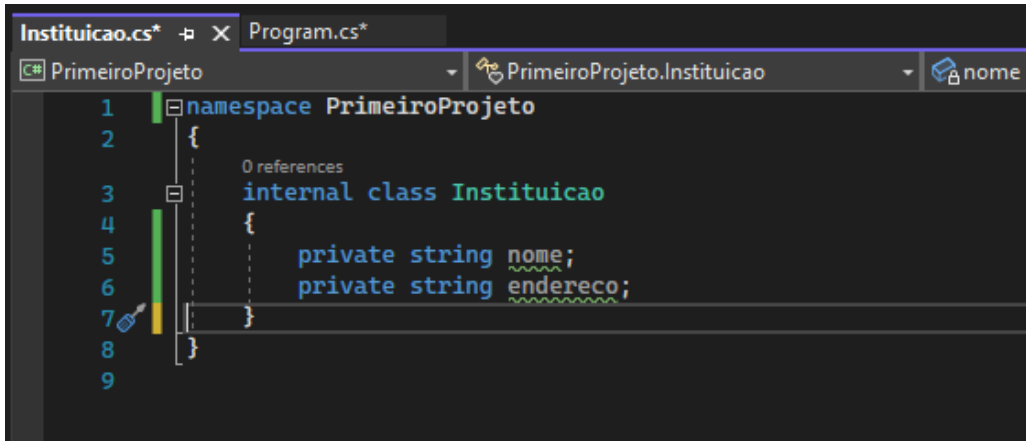


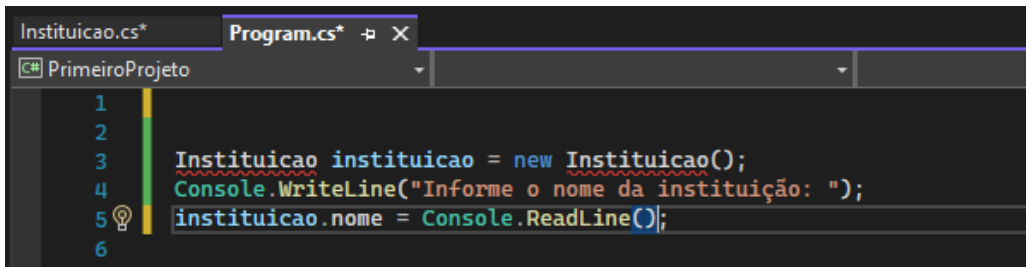
ANOTAÇÕES DO CAPÍTULO 02

Na Classe **Instituicao**, criar as strings **nome** e **endereco**:



```
1 namespace PrimeiroProjeto
2 {
3     0 references
4     internal class Instituicao
5     {
6         private string nome;
7         private string endereco;
8     }
9 }
```

Na classe **Program.cs**, que é o principal, fazer uma referência direta à variável **nome** definida na classe **Instituicao**.



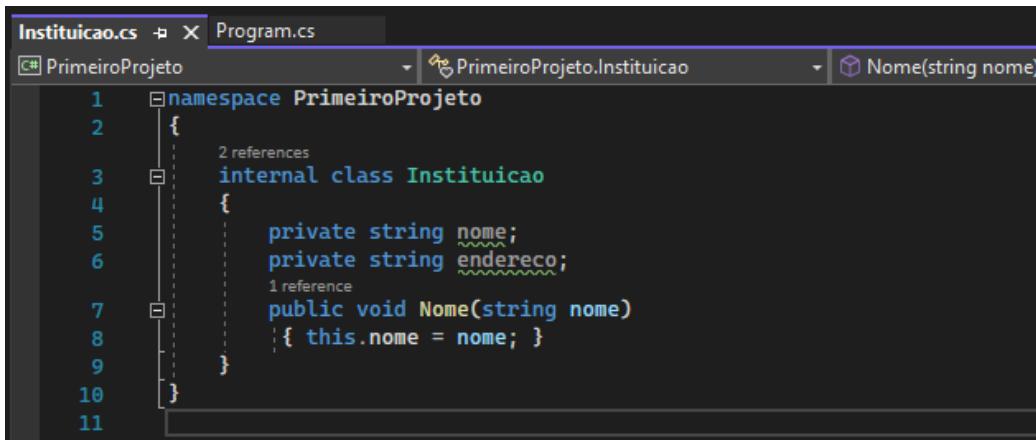
```
1
2
3 Instituicao instituicao = new Instituicao();
4 Console.WriteLine("Informe o nome da instituição: ");
5 instituicao.nome = Console.ReadLine();
6
```

MENSAGEM DE ERRO:

“Instituicao.nome é inacessível devido ao seu nível de proteção”

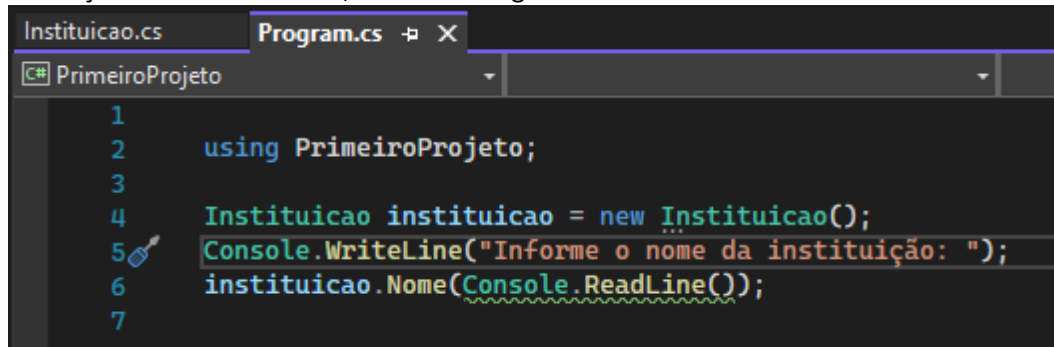
Um campo de uma classe não pode ser acessado diretamente de seus objetos, pois, para isto, precisamos de métodos que ofereçam este serviço.

Implementação do método **Nome**



```
1 namespace PrimeiroProjeto
2 {
3     2 references
4     internal class Instituicao
5     {
6         private string nome;
7         private string endereco;
8         1 reference
9         public void Nome(string nome)
10        { this.nome = nome; }
11    }
12 }
```

Utilização do método **Nome**, na classe Program.cs



```
1
2 using PrimeiroProjeto;
3
4 Instituicao instituicao = new Instituicao();
5 Console.WriteLine("Informe o nome da instituição: ");
6 instituicao.Nome(Console.ReadLine());
7
```

OBSERVAÇÃO

- Foi feito um método para a atribuição do valor.
- É necessário mais um método para a recuperação do valor.
- Neste caso seriam necessários dois métodos para que um único campo pudesse ter o seu valor atribuído e recuperado.

CAMEL CASE

é a denominação em [inglês](#) para a prática de escrever as palavras compostas ou [frases](#), onde cada palavra é iniciada com [maiúsculas](#) e unidas sem espaços.^[1] É um padrão largamente utilizado em diversas [linguagens de programação](#), como [Java](#), [C#](#), [Ruby](#), [PHP](#) e [Python](#), principalmente nas definições de [classes](#) e [objetos](#).^{[2][3]} Pela sua associação com [tecnologia](#), o marketing se apropriou dessa maneira de escrever, injetando certo ar de "tecnologia" nos produtos assim nomeados: [iPod](#), [eBay](#), [GameCube](#), [OpenOffice.org](#), [StarCraft](#), dentre outros.

A provável origem do termo é a semelhança do contorno de expressões CamelCase, onde as letras em maiúsculo "saltam" no meio das minúsculas como corcovas de um [camelo](#).

SOLUÇÃO

Nos casos de campos com acesso de escrita e leitura

```
namespace PrimeiroProjeto
{
    internal class Instituicao
    {
        public string Nome { get; set; }
        public string Endereco { get; set; }
    }
}
```

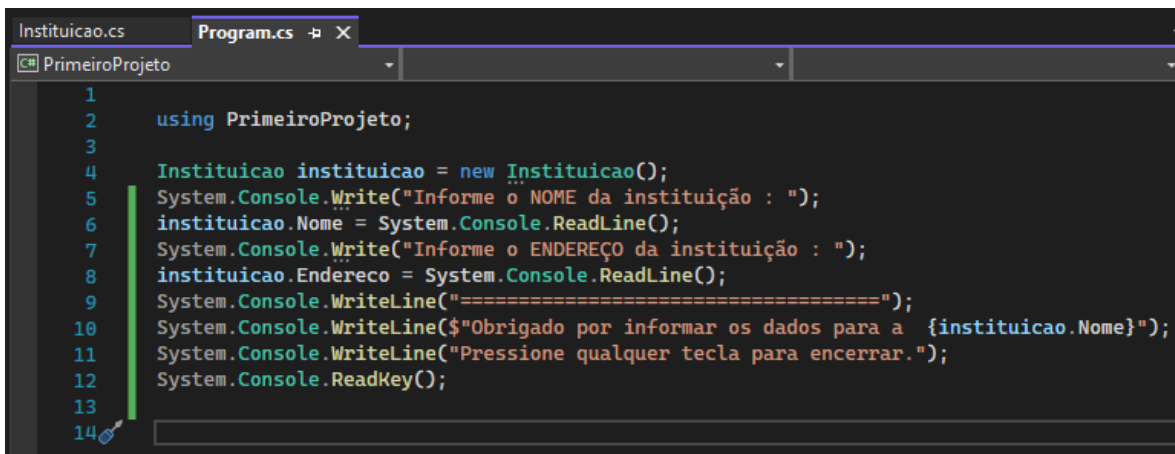
Transformar a declaração do campo em declaração de PROPRIEDADE. Nome e Endereco por convenção passam a ser grafados com letras maiúsculas.

OBSERVAÇÃO

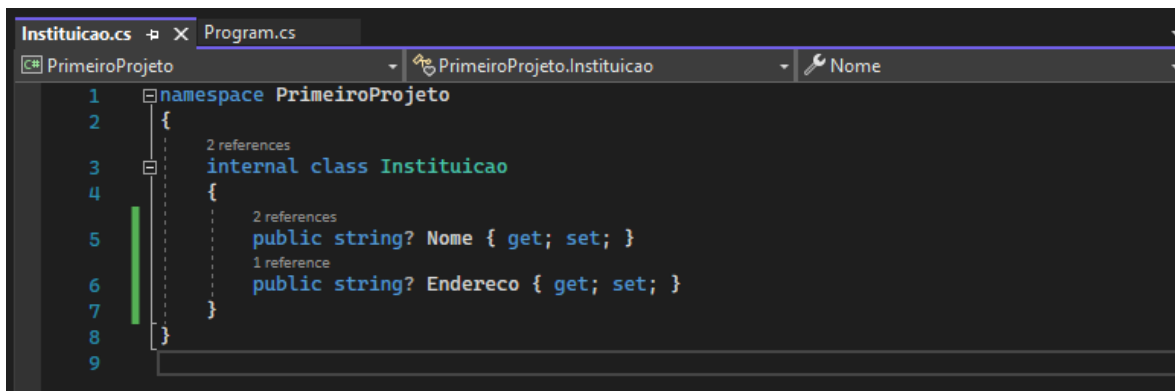
Atribuição de valor à propriedade:

```
namespace PrimeiroProjeto
{
    internal class Instituicao
    {
        public string Tipo { get; } = "Ensino Superior"
    }
}
```

IMPLEMENTAÇÃO FINAL DO EXEMPLO DO CAPÍTULO-02



```
Instituicao.cs Program.cs
PrimeiroProjeto
1
2 using PrimeiroProjeto;
3
4 Instituicao instituicao = new Instituicao();
5 System.Console.WriteLine("Informe o NOME da instituição : ");
6 instituicao.Nome = System.Console.ReadLine();
7 System.Console.WriteLine("Informe o ENDEREÇO da instituição : ");
8 instituicao.Endereco = System.Console.ReadLine();
9 System.Console.WriteLine("=====");
10 System.Console.WriteLine($"Obrigado por informar os dados para a {instituicao.Nome}");
11 System.Console.WriteLine("Pressione qualquer tecla para encerrar.");
12 System.Console.ReadKey();
13
14
```



```
Instituicao.cs Program.cs
PrimeiroProjeto PrimeiroProjeto.Instituicao Nome
1 namespace PrimeiroProjeto
2 {
3     2 references
4     internal class Instituicao
5     {
6         2 references
7         public string? Nome { get; set; }
8         1 reference
9         public string? Endereco { get; set; }
10 }
11
```

FIM DO CAPÍTULO 02