using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CursoCSharpBackup.Exercicios

{

    /\* Exercícios Teóricos

   1. Definição e Propósito:

    Pergunta: O que significa a palavra-chave var em C# e qual é o seu propósito?

    Resposta: var em C# significa inferencia de tipo. Ela permite que o compilador determine o tipo de uma variavel sem a necessidade de declarar explicitamente.

   2. Inferência de Tipo:

    Pergunta: Como o compilador determina o tipo de uma variavel quando voce usa var?

    Resposta: Quando voce usa var em C#, o compilador determina o tipo da variavel com base no valor que é atribuido a ela no momento da inicialização.

    O tipo é inferido a partir da expressão que voce usa para inicializar a variavel.

    O compilador realiza essa inferência no momento da compilação e uma vez que o tipo é inferido ele não pode ser alterado.

   3. Regras e Restrições:

    Pergunta: Quais são as restrições no uso da palavra-chave var em C#?

    Resposta: A palavra-chave var em C# tem algumas restrições importantes:

    Inicialização Obrigatoria: Ao usar var, a variavel deve ser inicializada na mesma linha de declaração. O compilador precisa do valor atribuido para inferir o tipo da variavel. Por exemplo:

    var idade = 30; // Correto: o tipo 'int' é inferido a partir do valor 30.

    var idade; // Incorreto: a variavel 'idade' não está inicializada, então o tipo não pode ser inferido.

    Inferência de Tipo Única: O tipo da variavel deve ser claramente inferido a partir do valor atribuido. Se o tipo não puder ser determinado de maneira inequívoca, o código não compilará.

    Não Pode Ser Usado Sem Inicialização: var não pode ser usado para declarar uma variavel sem inicializá-la. Além disso, var não pode ser usado para declarar variáveis de tipo anônimo fora de uma expressão de inicialização.

   4. Diferença entre var e Tipos Explícitos:

    Pergunta: Qual é a diferença entre usar var e declarar uma variavel com um tipo explícito?

    Resposta: var

    Inferência de Tipo: O tipo da variável é automaticamente determinado pelo compilador com base no valor atribuído a ela.

    Sintaxe Simples: Reduz a quantidade de código necessário, tornando-o mais conciso.

    Requisitos: A variável precisa ser inicializada na declaração para que o tipo possa ser inferido.

    Tipo Explícito

    Declaração Clara: O tipo da variável é especificado diretamente na declaração.

    Controle Preciso: Dá controle exato sobre o tipo da variável, o que pode melhorar a clareza e a segurança do tipo.

    Flexibilidade: Não depende da inicialização para determinar o tipo; o tipo é explicitamente declarado.

    \*/

    /\* Declaração de Variáveis com var:

    Exercício: Declare uma variável de cada um dos seguintes tipos usando var e atribua a elas valores apropriados. Depois, exiba o tipo da variável usando GetType() para confirmar a inferência do tipo. \*/

    class Program

    {

        static void Main()

        {

            var inteiro = 10;  // Tipo inferido como int

            var texto = "Olá, Mundo!";  // Tipo inferido como string

            var flutuante = 3.14f;  // Tipo inferido como float

            var duplo = 3.14;  // Tipo inferido como double

            var booleano = true;  // Tipo inferido como bool

            Console.WriteLine(inteiro.GetType());  // Output: System.Int32

            Console.WriteLine(texto.GetType());  // Output: System.String

            Console.WriteLine(flutuante.GetType());  // Output: System.Single

            Console.WriteLine(duplo.GetType());  // Output: System.Double

            Console.WriteLine(booleano.GetType());  // Output: System.Boolean

        }

    }

    /\* Utilizando var em Métodos de LINQ:

    Exercício: Crie uma lista de inteiros e use um método LINQ para ordenar a lista em ordem crescente. Declare a variável que armazena o resultado usando var. \*/

    class Program

    {

        static void Main()

        {

            var lista = new List<int> { 5 , 3 , 8 , 1 , 2 };

            var listaOrdenada = lista.OrderBy(x => x).ToList();

            Console.WriteLine("Lista ordenada:");

            foreach (var item in listaOrdenada)

            {

                Console.WriteLine(item);

            }

        }

    }

    /\* Declaração de Variáveis com Tipos Complexos:

    Exercício: Declare uma variável var para um tipo complexo, como um Dictionary ou uma instância de uma classe personalizada. Mostre como a inferência de tipo funciona com tipos mais complexos. \*/

    class Program

    {

        static void Main()

        {

            // Usando Dictionary

            var dicionario = new Dictionary<string, int>

            {

                { "Um", 1 },

                { "Dois", 2 },

                { "Três", 3 }

            };

            Console.WriteLine("Dicionário:");

            foreach (var item in dicionario)

            {

                Console.WriteLine($"{item.Key}: {item.Value}");

            }

        }

    }

    /\* Refatoração de Código:

    Exercício: Refatore o seguinte código para usar var em vez de tipos explícitos onde possível. \*/

    /\* Trabalhando com Tipos Anônimos: Exercício: Crie um tipo anônimo usando var para armazenar informações de uma pessoa (nome e idade) e exiba essas informações. \*/

    class Program

    {

        static void Main()

        {

            // Criando um tipo anônimo para armazenar informações de uma pessoa

            var pessoa = new

            {

                Nome = "Ana",

                Idade = 30

            };

            // Exibindo as informações armazenadas no tipo anônimo

            Console.WriteLine($"Nome: {pessoa.Nome}");

            Console.WriteLine($"Idade: {pessoa.Idade}");

        }

    }

}