Funções locais em C#.

Jonathan Vinícius Suter

1 Para que servem as funções?

As funções e/ou procedimentos possuem o fim de centralizar trechos de código para permitir a reutilização dos mesmos e evitar que se reescreva códigos repetidos para o mesmo tipo de operações no mesmo contexto, além de facilitar a manutenção da aplicação.

2 Utilização

2.1 Como escrever uma função

Para a escrita de uma função, a mesma deve estar dentro do escopo de um bloco de código com a diretiva `class`, e começar com o texto `static`, depois, o tipo de retorno (se será do tipo `int`, `double`, `char`, `string`, ou qualquer outro tipo dentro da linguagem) e o nome da mesma, que deve ser único dentro do escopo de `class`

```
class Program
{
        //a própria “Main” é uma função que é identificada como a partida de um programa de
console no c#. Ela não possui retorno e apenas é executada:
        static void Main(string[] args)
        {
            //...execução de código...
        }

}
```

2.2 Funções sem retorno (procedimentos)

Uma função sem retorno indica uma execução de procedimento em que **não** há uma devolução de valores. Ela é caracterizada pelo uso do termo `void` que significa “vazio”.

```
class Program
{
        static void Main(string[] args)
        {
            //com passagem de parâmetros.
            MostrarUmTexto("É um texto aleatório.");
            SomarDoisNumeros(10.5, 20.3);
            //sem passagem de parâmetros.
            MultiplicarDoisValoresEstaticos();
        }
        static void MostrarUmTexto(string texto)
        {
            Console.WriteLine(texto);
        }
        static void SomarDoisNumeros(double n1, double n2)
        {
            double res = n1 + n2;
            Console.WriteLine("A Soma é "+res);
        }
        static void MultiplicarDoisValoresEstaticos()
        {
            double res = 20 * 50.5;
            Console.WriteLine("A multiplicação é"+res);
        }
}
```

2.3 Funções com retorno:

Uma função que possui retorno, pode receber parâmetros (ou não) executar uma ação e devolver valores de um tipo específico:

```
class Program
{
        static void Main(string[] args)
        {
            //a função RetornarTextoParamostrar recebe um parâmetro do tipo string, esta
fará o processamento e retornará o valor (do tipo string).
            string texto = RetornarTextoParamostrar("Hello World!");
            Console.WriteLine(texto);

            PedirEntradaParaSomaDeDoisNumeros();

            //pede ao usuário que informe um texto para processamento;
            Console.WriteLine("Insira um texto para quebrar");
            string entrada = Console.ReadLine();

            //retorna a string como um array de char
            char[] retorno = QuebrarString(entrada);

            //imprime na tela o retorno
            foreach (char item in retorno)
            {
                Console.WriteLine(item);
            }
            entrada = MontarString(retorno);
            Console.WriteLine("vetor de char reconvertido em String: " + entrada);
        }
        //esta função retorna um texto, interpolado com o parâmetro passado.
        static string RetornarTextoParamostrar(string texto)
        {
            return $"Texto para retornar na principal: {texto}";
        }
```

```
        //Soma dois números e retorna o valor total.
        static int SomarNumeros(int a, int b)
        {
            return a + b;
        }
        //Mesma função que SomarNumeros, porém, soma valores diferentes.
        //O compilador entende que, por receberem valores de tipos diferentes, têm assina-
turas diferentes e são diferentes.
        static double SomarNumeros(double a, double b)
        {
            return a + b;
        }
        //esta função apenas executa um bloco de operações e não possui retorno.
        static void PedirEntradaParaSomaDeDoisNumeros()
        {
            int num1;
            int num2;
            double num1d;
            double num2d;
            Console.WriteLine("Insira um valor para num1");
            num1 = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Insira um valor para num2");
            num2 = Convert.ToInt32(Console.ReadLine());
            //a linguagem infere que é a função com retorno de inteiro
            int retorno = SomarNumeros(num1, num2);
            Console.WriteLine("O valor da soma é: " + retorno);

            Console.WriteLine("Insira um valor para num1d");
            num1d = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine("Insira um valor para num2d");
            num2d = Convert.ToDouble(Console.ReadLine());

            //é possível chamar uma função dentro de outra
            double retornoDouble = SomarNumeros(num1d, num2d);
            Console.WriteLine("O valor da soma é: " + retornoDouble);
        }
        //quebra uma string em um vetor de char
        static char[] QuebrarString(string texto)
        {
            char[] textoQuebrado = new char[texto.Length];
            for (int i = 0; i < texto.Length; i++)
            {
                textoQuebrado[i] = texto[i];
            }
            return textoQuebrado;
        }
        static string MontarString(char[] texto)
        {
            StringBuilder retorno = new StringBuilder();
            foreach (char item in texto)
            {
                retorno.Append(item);
            }
            return retorno.ToString();
        }
}
```

3 Exercícios

Faça um programa com um menu que receba uma opção, e de acordo com a opção, efetue uma operação (uma função ou procedimento) conforme abaixo e mostre em tela o resultado. O usuário deve poder efetuar quantas operações quiser.

1 – Adição;

2 – Subtração;

3 – Multiplicação;

4 – Divisão (não permitir divisão por zero);

0 – Sair.

Faça um programa em que o usuário preencha um array de 10 posições e mostre o maior valor entre eles (usando funções e/ou procedimentos).

Faça um programa que receba um valor e, usando uma função, receba esse valor e retorne e mostre todos os divisores entre 0 e o número;

Faça uma função que receba em um vetor as notas de 10 alunos e verifique se todos passaram (nota maior que 7). Deverá ser criada uma função que retorne um *array* do tipo *bool*, com os alunos que passaram.