

# Programação C# + ASP.NET

**Prof. Me. Daniel Menin Tortelli**

**e-mail:** [danielmenintortelli@gmail.com](mailto:danielmenintortelli@gmail.com)

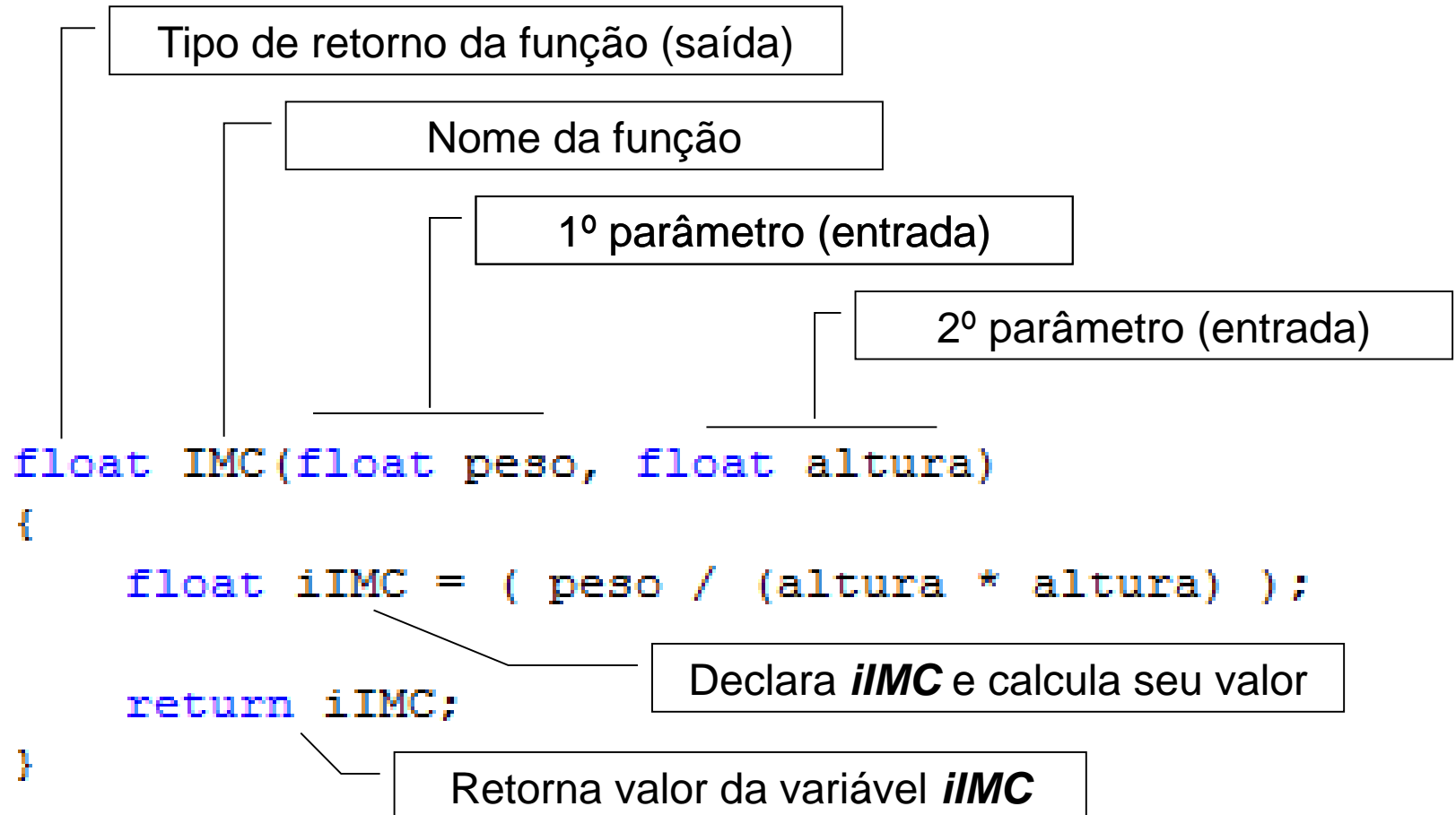
**Site:** <http://sites.google.com/site/danielmenintortelli/home>

# Funções

# Funções

- A maioria dos programas de computador que resolvem problemas do mundo real são muito maiores do que os programas apresentados até agora.
- A experiência tem mostrado que a melhor maneira de desenvolver e manter um programa grande é construí-lo a partir de pequenas partes ou componentes, sendo cada uma delas mais fácil de manipular que o programa original.
- Essa técnica é chamada de ***dividir para conquistar***.
- Os módulos em C# são chamados de *funções* e *classes*.
- Os programas em C# são escritos tipicamente combinando-se funções novas que o programador escreve com “funções pré-empacotadas” disponíveis na *biblioteca padrão de C#* ou outras bibliotecas...

# Exemplo de Função: Cálculo do IMC (Índice de Massa Corporal)



# Trabalhando com Funções

```
class Program
{
    // função que calcula soma de dois inteiros
    static int CalculaSoma(int num1, int num2)
    {
        int soma = (num1 + num2);

        return soma;
    }

    // função que calcula subtração de dois inteiros
    static int CalculaSubtracao(int num1, int num2)
    {
        int soma = (num1 - num2);

        return soma;
    }
}
```

# Trabalhando com Funções

```
// Função Principal
static void Main(string[] args)
{
    int iNum1 = 0;
    int iNum2 = 0;

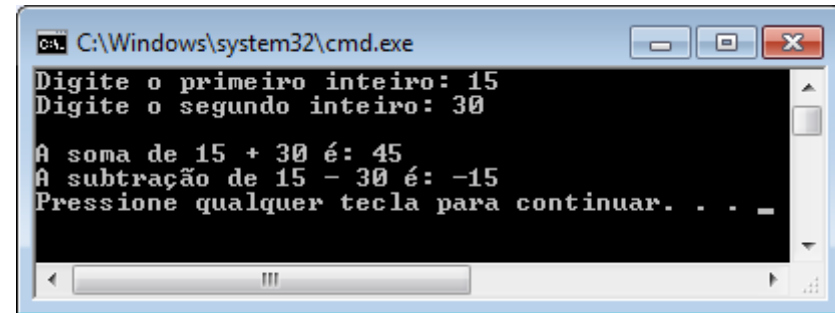
    Console.Write("Digite o primeiro inteiro: ");
    iNum1 = Convert.ToInt32(Console.ReadLine());

    Console.Write("Digite o segundo inteiro: ");
    iNum2 = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine(); // linha em branco

    Console.WriteLine("A soma de {0} + {1} é: {2}", iNum1, iNum2, CalculaSoma(iNum1, iNum2));
    Console.WriteLine("A subtração de {0} - {1} é: {2}", iNum1, iNum2, CalculaSubtracao(iNum1, iNum2));

    } // fim função Main
} // fim classe Program
```



# Exercício

- Modifique o programa anterior fazendo com que os valores aceitos pelo programa sejam do tipo ***float***.
- Exiba o resultado dos cálculos com 2 casas decimais.

# Cálculo do IMC – Programa

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace CalculoIMC
{
    class Program
    {
        static float CalcularIMC(float peso, float altura)
        {
            float fIMC = (peso / (altura * altura));

            return fIMC;
        }
    }
}
```



# Cálculo do IMC – Programa

```
static void ClassificarIMC(float imc)
{
    Console.WriteLine(); // linha em branco

    if (imc < 20)
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está abaixo da faixa considerada normal!");
    }
    else if ( (imc >= 20) && (imc < 25) )
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está dentro da faixa considerada normal!");
    }
    else if ( (imc >= 25) && (imc < 30) )
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está na faixa considerada excesso de peso!");
    }
    else if ( (imc >= 30) && (imc < 35) )
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está na faixa considerada obesidade leve!");
    }
    else if ( (imc >= 35) && (imc < 40) )
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está na faixa considerada obesidade moderada!");
    }
    else // se for maior do que 40
    {
        Console.WriteLine("Seu IMC é: {0:f1}", imc);
        Console.WriteLine("Seu peso está na faixa considerada obesidade mórbida!");
    }
}
```

# Cálculo do IMC – Programa

```
static void Main(string[] args)
{
    float fPeso = 0.0f;
    float fAltura = 0.0f;
    float fIMC = 0.0f;

    Console.Write("Digite seu peso em Kg (ex: 75,5): ");
    fPeso = Convert.ToSingle(Console.ReadLine());

    Console.Write("Digite sua altura em metros (ex: 1,60): ");
    fAltura = Convert.ToSingle(Console.ReadLine());

    fIMC = CalcularIMC(fPeso, fAltura);

    ClassificarIMC(fIMC);
} // fim função Main
} // fim classe Program
} // fim namespace CalculoIMC
|
```

# Funções de Conversão – Programa

```
namespace ProgramaConversao
{
    class Program
    {
        // Esta função converte um valor em metros para centímetros
        static float MetrosParaCentimetros(float metros)
        {
            float fResultado = 0.0f;

            fResultado = (metros * 100);

            return fResultado;
        }

        // Esta função converte um valor em metros para milímetros
        static float MetrosParaMilimetros(float metros)
        {
            float fResultado = 0.0f;

            fResultado = (metros * 1000);

            return fResultado;
        }
    }
}
```

# Funções de Conversão – Programa

```
// Função principal
static void Main(string[] args)
{
    // Variável que armazenará a opção do menu
    // escolhida pelo usuário
    int iMenuOP = 0;

    // Imprime menu de opções de conversão
    Console.WriteLine("-----");
    Console.WriteLine("| Escolha a opção para Conversão |");
    Console.WriteLine("-----");
    Console.WriteLine(" 1 - Metros -> Centímetros");
    Console.WriteLine(" 2 - Metros -> Milímetros");
    Console.WriteLine(" 0 - Sair do Programa");

    // Solicita que o usuário selecione uma opção
    iMenuOP = Convert.ToInt32(Console.ReadLine());
}
```

```
// Analisa a escolha do usuário
```

```
switch (iMenuOP)
```

```
{
```

```
    case 0:
```

```
        break;
```

```
    case 1:
```

```
    {
```

```
        float fValorConvertido, fMetros = 0.0f;
```

```
        Console.Write("Digite um valor em metros:");
```

```
        fMetros = Convert.ToSingle(Console.ReadLine());
```

```
        fValorConvertido = MetrosParaCentimetros(fMetros);
```

```
        Console.WriteLine("Metros: {0:f1} -> Centimetros: {1:f1}", fMetros, fValorConvertido);
```

```
    }
```

```
    break;
```

```
    case 2:
```

```
    {
```

```
        float fValorConvertido, fMetros = 0.0f;
```

```
        Console.Write("Digite um valor em metros:");
```

```
        fMetros = Convert.ToSingle(Console.ReadLine());
```

```
        fValorConvertido = MetrosParaMilimetros(fMetros);
```

```
        Console.WriteLine("Metros: {0:f1} -> Milimetros: {1:f1}", fMetros, fValorConvertido);
```

```
    }
```

```
    break;
```

```
    default:
```

```
        break;
```

```
    } // fim switch
```

```
    } // fim função MAIN
```

```
    } // fim classe Program
```

```
} // fim namespace ProgramaConversao
```

# ref (Referência de C#)

- A palavra-chave de **ref** causa um argumento a ser passado por referência, não por valor.
- O efeito de passagem por referência é que qualquer alteração no parâmetro no método é refletida na variável subjacente o argumento o método de chamada.
- O valor de um parâmetro de referência é sempre o mesmo que o valor da variável subjacente ao argumento.
- Um argumento que é passado em um parâmetro de **ref** deve ser inicializado antes de ser passado.
- Uma variável de um tipo de referência não contém seus dados diretamente. Ele contém uma referência a seus dados.

# ref (Referência de C#)

```
static void PassaPorValor(int val)
{
    val = val + 44;
}

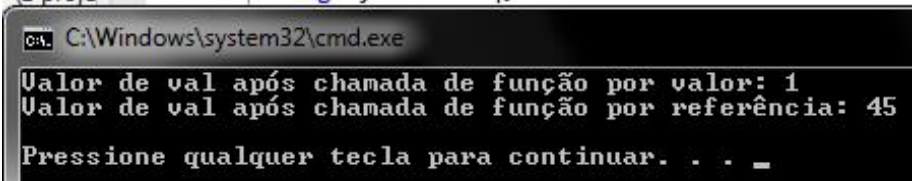
static void PassaPorReferencia(ref int val)
{
    val = val + 44;
}
```

```
static void Main(string[] args)
{
    int val = 1; // Variável global

    PassaPorValor(val); // Passa val para a função por valor
    Console.WriteLine("Valor de val após chamada de função por valor: {0}", val);

    PassaPorReferencia(ref val); // Passa val para a função por referência
    Console.WriteLine("Valor de val após chamada de função por referência: {0}", val);

    Console.WriteLine(); // linha em branco
}
```



```
C:\Windows\system32\cmd.exe
Valor de val após chamada de função por valor: 1
Valor de val após chamada de função por referência: 45
Pressione qualquer tecla para continuar. . . _
```

# Funções de Conversão – Exercício 1

- Melhorar o programa anterior, criando novas funções de conversão:
  1. Crie uma função que converta temperatura de Celsius para Fahrenheit.
  2. Crie uma função que converta temperatura de Fahrenheit para Celsius.
  3. Crie uma função que converta Quilos para Gramas.
  4. Crie uma função que converta Gramas para Quilos.



## Exercícios 2

Crie um programa que possua uma função chamada ***Calculidade(int anoNasc)*** que retorna a idade do usuário.

A função tem como parâmetro de entrada apenas o ano de nascimento do usuário.

Exiba a idade do usuário.

## Exercícios 3

Crie um programa que possua um método chamado ***Distance*** para calcular e exibir a distância entre dois pontos (x1, y1) e (x2, y2), segundo o Teorema de Pitágoras. Todos os números e valores de retorno devem ser do tipo ***double***. O programa deve solicitar ao usuário a inserção das coordenadas dos pontos e passar esses valores como parâmetros da função:

***DistanciaEntrePontos(double x1, double y1, double x2, double y2)***

$$D = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$$

# Exercícios 4

- Uma empresa quer verificar se um empregado está qualificado para a aposentadoria ou não. Para estar em condições, um dos seguintes requisitos deve ser satisfeito:
  - Ter no mínimo 65 anos de idade.
  - Ter trabalhado no mínimo 30 anos.
  - Ter no mínimo 60 anos e ter trabalhado no mínimo 25 anos.
- Com base nas informações acima, faça um algoritmo que leia: o número do empregado (código), o ano de seu nascimento e o ano de seu ingresso na empresa.
- O programa deverá exibir a idade e o tempo de trabalho do empregado e a mensagem 'Requerer aposentadoria' ou 'Não requerer'.

## Exercícios 5

Escreva um programa em C# para solicitar ao usuário o raio (do tipo **double**) de uma esfera e, chame a função **SphereVolume** para calcular e exibir o volume da esfera. Utilize a seguinte equação para calcular o volume da esfera:

```
double volume = ( 4.0 / 3.0 ) * Math.PI * Math.pow( radius, 3 )
```

Escreva um programa em C# com uma função chamada

***IntegerPower( base, exponent )*** que retorna o valor de:

***base<sup>exponent</sup>***

## Exercícios 6

Escreva um programa em C# com uma função chamada

***IntegerPower( base, exponent )*** que retorna o valor de:

***base<sup>exponent</sup>***

# Exercícios 7

- Faça um algoritmo para ler: número da conta do cliente, saldo, débito e crédito.
- Após, crie uma função para calcular e escrever o saldo atual ( $\text{saldo atual} = \text{saldo} - \text{débito} + \text{crédito}$ ).
- Também testar se saldo atual for maior ou igual a zero escrever a mensagem 'Saldo Positivo', senão escrever a mensagem 'Saldo Negativo'.

## Exercícios 8

- Escreva um algoritmo que leia o número de litros vendidos e o tipo de combustível (*codificado da seguinte forma: **A**-álcool, **G**-gasolina*).
- Crie uma função que calcule e imprima o valor a ser pago pelo cliente sabendo-se que o preço do litro da gasolina é R\$ 3,30 e o preço do litro do álcool é R\$ 2,90.

# Exercícios 9

- Faça um algoritmo para ler: a descrição do produto (nome), a quantidade adquirida e o preço unitário.
- Crie uma função para calcular e escrever o total (total = quantidade adquirida \* preço unitário), o desconto e o total a pagar (total a pagar = total - desconto), sabendo-se que:
  - Se quantidade  $\leq 5$  o desconto será de 2%
  - Se quantidade  $> 5$  e quantidade  $\leq 10$  o desconto será de 3%
  - Se quantidade  $> 10$  o desconto será de 5%



## Exercícios 10

- Solicitar ao usuário a entrada de 2 valores, referentes ao **raio** e **altura** de um cilindro. Em seguida:
  - a) Chamar a função **CalculaAreaLateral** para calcular e mostrar a área lateral do cilindro;
  - b) Chamar a função **CalculaAreaTotal** para calcular e mostrar a área total do cilindro;
  - c) Chamar a função **CalculaVolume** para calcular e mostrar o volume do cilindro;

$$A(\text{base}) = \pi \times r^2$$

$$A(\text{lateral}) = 2 \times \pi \times r \times h$$

$$A(\text{total}) = A(\text{lateral}) + 2 \times A(\text{base})$$

$$\text{Volume} = A(\text{base}) \times h$$