

# C# - C Sharp

## AULA 07

### ROTINAS – PROCEDIMENTOS E FUNÇÕES

**Prof. Emanuel Braz da Cruz**

#### INTRODUÇÃO

As rotinas, que são as funções e os procedimentos, constituem um conceito fundamental tanto na programação tradicional, denominada de procedural, quanto também na programação orientada a objetos que, como já estudado, constituem os métodos que determinam e alteram o comportamento dos objetos.

Nestes ambientes visuais, com uma IDE semelhante ao Visual Studio, o Delphi, Eclipse (para o Java e outras linguagens), etc., o processamento pode ser implementado no próprio evento clique. Neste mesmo evento, pode também implementar a chamada de uma função ou procedimento e, é claro, deve implementar também a rotina (procedimento ou função) que é chamada deste evento.

Conforme aprendido nas linguagens básicas, a função sempre retorna valor. Portanto, ao chamar uma função de um evento clique, ou de uma outra rotina, o valor a ser retornado pela função deverá forçosamente retornar para o evento clique ou para a rotina que chamou a função. Este conceito é sempre desta forma e jamais muda.

Como também é do conhecimento básico, o procedimento não retorna valor. No entanto, as linguagens de programação possuem um recurso que através de variáveis passadas por parâmetros, podem passar um valor calculado por um procedimento para o evento clique ou a rotina que chamou o procedimento.

Desta forma, com um procedimento é possível também ter um valor calculado no procedimento e este valor é passado (evito a palavra retornado, quem retorna valor é a função) para o evento clique ou uma rotina, que chama o procedimento. Além deste recurso de passar valor para a rotina ou evento clique, o procedimento pode também apresentar ou exibir no próprio procedimento um valor por ele, procedimento, calculado ou processado.

Sendo assim, com uma função é possível afirmar que ela sempre retorna valor. Devido a este motivo, a função não exibe ou mostra o resultado processado na função. Mas o procedimento, através de variáveis passadas por parâmetros, passa valor processado para a chamada do procedimento. E, não esqueça, pode também no próprio procedimento exibir ou mostrar o resultado do processamento. Portanto, são duas metodologias de processamento com procedimento: exibindo ou mostrando o resultado do processamento no próprio procedimento ou passando variáveis por parâmetro (processado pelo procedimento) para o evento clique ou rotina que chamou o procedimento.

As rotinas, segundo o conceito da análise de sistemas, devem ser projetadas seguindo dois conceitos que são acoplamento e coesão. Estes dois conceitos especificam que uma rotina deve ter os parâmetros suficientes e necessários para que a rotina atinja seu objetivo e este objetivo deve ser único, bem determinado e não subdividido em outros.

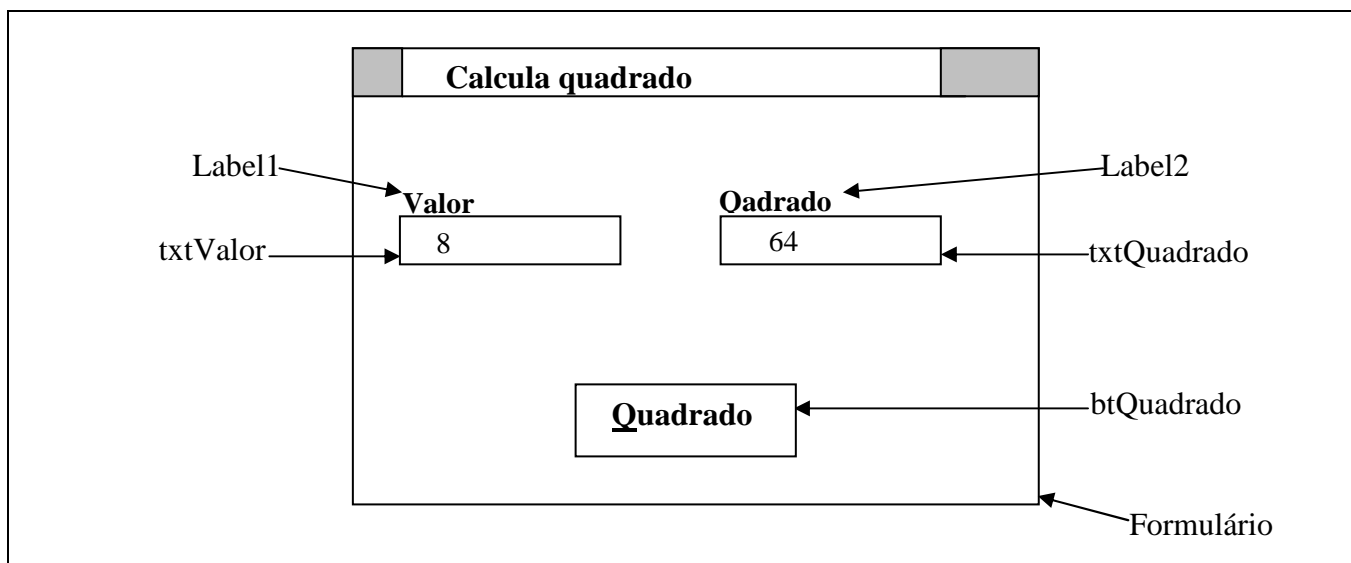
## PROCESSAMENTO NO PRÓPRIO EVENTO CLIQUE

No intuito de melhor atender os conceitos envolvendo a programação com rotinas, funções e procedimentos, é recomendável que seja apresentado um processamento sem estes recursos, efetuando o processamento do que é solicitado no próprio evento clique. Desta forma, nenhuma rotina é chamada, e no próprio evento clique é feito o cálculo ou o processamento desejado. O programa a seguir implementa o cálculo solicitado no próprio evento clique. Lembre-se que não há chamada de rotina (função ou procedimento) neste exercício. Para obter a solução o exemplo processa o cálculo no próprio evento click.

### 1. Exercício proposto:

Implementar um programa que obtém um valor numérico de um objeto da classe TextBox, denominado de txtValor, calcular o seu quadrado, e exibir o resultado em um outro objeto denominado de txtQuadrado, também pertencente à classe TextBox. O cálculo do quadrado solicitado deverá ocorrer quando o usuário clicar no botão btQuadrado. O cálculo deverá ser efetuado no próprio evento clique.

Antes de implementar o programa é necessário pensar na interface (sempre pensar na interface) e quais os objetos que vão compor o formulário. Criar um novo projeto e adicionar no formulário os objetos não existentes com suas respectivas propriedades, conforme as tabelas que se encontram a seguir, após a figura do layout do formulário.



Abaixo se encontram os objetos que deverão fazer parte da interface, que é o formulário, e a configuração destes objetos que deverão estar contidos no formulário:

#### Objeto Form1 da Classe System.Windows.Forms.Form

PROPRIEDADE	VALOR	DESCRIÇÃO
-------------	-------	-----------

<b>Name</b>	frmCalculaQuadrado	Já descrito.
<b>Text</b>	Processa no evento clique	Já descrito.
<b>Ico</b>	FLGBRAZL.ICO	Indica o arquivo de extensão .ICO para o formulário. Esta imagem é mostrada na caixa de menu do sistema do formulário e quando o formulário é minimizado.
<b>MaximizeBox</b>	False	Desabilita o ícone de maximizar o formulário.
<b>FormBorderStyle</b>	FixedSingle	Indica a aparência e comportamento da borda e a barra de título do formulário. O valor FixedSingle não permite que o usuário altere a largura e altura do formulário.
<b>Font</b>	Bold	Todos os objetos contidos no formulário passam ter texto em negrito.
<b>StartPosition</b>	CenterScreen	Determina a posição do formulário quando ele é apresentado na tela desktop. A constante CenterScreen faz com que o formulário seja centralizado na tela (Screen).

#### Objeto Button1 da Classe System.Windows.Forms.Button

PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	btQuadrado	Já descrito.
<b>Text</b>	&Processa	Já descrito.

#### Objeto Label1 da Classe System.Windows.Forms.Label

PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	Label1	Já descrito.
<b>Text</b>	Valor	Já descrito.

#### Objeto Label2 da Classe System.Windows.Forms.Label

PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	Label2	Já descrito.
<b>Text</b>	Quadrado	Já descrito.

#### Objeto Text1 da Classe System.Windows.Forms.TextBox

PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	txtValor	Já descrito.
<b>Text</b>		Permite digitar ou obter uma string que se encontra como conteúdo da caixa de texto.

#### Objeto Text1 da Classe System.Windows.Forms.TextBox

PROPRIEDADE	VALOR	DESCRIÇÃO
-------------	-------	-----------

Name	txtQuadrado	Já descrito.
------	-------------	--------------

O código a seguir implementa o processamento solicitado que consiste de, no evento clique, obter o valor contido em uma caixa de texto, denominada de txtValor, e apresentar o quadrado desse valor em uma outra caixa de texto denominada de txtQuadrado, quando o usuário clicar no botão denominado de btQuadrado.

```

1 using System.ComponentModel;
2 using System.Data;
3 using System.Drawing;
4 using System.Linq;
5 using System.Text;
6 using System.Windows.Forms;
7
8 namespace ProcessaNoEventoClique
9 {
10     public partial class frmProcessaNoEventoClique : Form
11     {
12         public frmProcessaNoEventoClique()
13         {
14             InitializeComponent();
15         }
16
17         private void btQuadrado_Click(object sender, EventArgs e)
18         {
19             txtQuadrado.Text = Convert.ToString(Convert.ToDecimal(txtValor.Text) *
20                                     Convert.ToDecimal(txtValor.Text));
21         }
22     }
23 }
24 }

```

É necessário ter muita atenção na solução do exercício anterior, comparando com os demais. Observe que o conteúdo da caixa de texto, txtValor, que é uma string (a propriedade Text é uma variável do tipo de dado string), é convertido para um valor numérico, decimal. Após a conversão para numérico, é efetuado o produto. Como o produto de dois valores numéricos tem como resultado um valor numérico do mesmo tipo de dado, decimal, é necessário transformar para string o resultado obtido para que o mesmo possa ser armazenado na caixa de texto, através da propriedade Text, que é do tipo de dado string. Ou seja, os tipos de dados têm sempre que ser compatíveis.

## 2. Exercício proposto:

Implementar um programa que obtenha quatro valores numéricos de objetos da classe TextBox, denominados de txtValor1, txtValor2, txtValor3 e txtValor4, e calcular a soma dos conteúdos numéricos destas caixas de textos. Exibir o resultado em um outro objeto denominado de txtSoma, também pertencente à classe TextBox. O cálculo da soma solicitada deverá ocorrer quando o usuário clicar no botão btSoma. O cálculo deverá ser efetuado no próprio evento clique.

Abaixo se encontram os objetos que deverão fazer parte da interface, que é o formulário, e a configuração destes objetos que deverão estar contidos no formulário:

**Objeto Form1 da Classe System.Windows.Forms.Form**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	frmCalculaQuadrado	Já descrito.
Text	Processa no evento clique	Já descrito.
Ico	FLGBRAZL.ICO	Já descrito.
MaximizeBox	False	Já descrito.
FormBorderStyle	FixedSingle	Já descrito.
Font	Bold	Já descrito.
StartPosition	CenterScreen	Já descrito.

**Objeto Button1 da Classe System.Windows.Forms.Button**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	btQuadrado	Já descrito.
Text	&Processa	Já descrito.

**Objeto Label1 da Classe System.Windows.Forms.Label**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	Label1	Já descrito.
Text	Valor1	Já descrito.

**Objeto Label1 da Classe System.Windows.Forms.Label**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	Label2	Já descrito.
Text	Valor2	Já descrito.

**Objeto Label1 da Classe System.Windows.Forms.Label**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	Label3	Já descrito.
Text	Valor3	Já descrito.

**Objeto Label1 da Classe System.Windows.Forms.Label**

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	Label4	Já descrito.
Text	Valor4	Já descrito.

**Objeto Label2 da Classe System.Windows.Forms.Label**

PROPRIEDADE	VALOR	DESCRIÇÃO
-------------	-------	-----------

<b>Name</b>	Label5	Já descrito.
<b>Text</b>	Soma	Já descrito.

#### Objeto Text1 da Classe System.Windows.Forms.TextBox

.PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	txtSoma	Já descrito.
<b>Text</b>		Permite digitar ou obter uma string que se encontra como conteúdo da caixa de texto.

#### Objeto Text1 da Classe System.Windows.Forms.TextBox

.PROPRIEDADE	VALOR	DESCRIÇÃO
<b>Name</b>	txtSoma	Já descrito.

A implementação da soma dos conteúdos das caixas de texto, após serem passados para numéricos, encontra-se abaixo:

```

1 using System;
2 using System.Collections.Generic;
5 using System.ComponentModel;
6 using System.Data;
7 using System.Drawing;
8 using System.Linq;
9 using System.Text;
10 using System.Windows.Forms;
11 namespace ProcessaSomaEventoClique
12 {
13     public partial class frmCalculaSoma : Form
14     {
15         public frmCalculaSoma()
16         {
17             InitializeComponent();
18         }
19
20         private void btSoma_Click(object sender, EventArgs e)
21         {
22             txtSoma.Text = Convert.ToString(Convert.ToDecimal( txtValor1.Text) +
23                                     Convert.ToDecimal( txtValor2.Text) +
24                                     Convert.ToDecimal( txtValor3.Text) +
25                                     Convert.ToDecimal( txtValor4.Text));
26         }
27     }
28 }

```

Observe no exercício anterior os passos necessários para efetuar o processamento desejado:

1. Converte para decimal (numérico) o conteúdo de cada caixa de texto, através do método Convert.

2. Lembre-se que os valores foram digitados em um objeto da classe TextBox. Estes valores digitados foram obtidos através da propriedade Text, que é do tipo de dado string e, com este tipo de dado, não se pode somar matematicamente.
3. Após converter o conteúdo de cada caixa de texto para numérico, decimal, efetua a soma dos valores numéricos obtidos com a conversão de string para decimal.
4. Após efetuar a soma dos valores numéricos do tipo de dado decimal, obtém-se um valor numérico, do mesmo tipo de dado, decimal.
5. Com o valor numérico obtido, que é o resultado desejado, tem-se que armazená-lo na caixa de texto que no formulário vai conter o resultado para que o usuário possa ver.
6. Este valor numérico obtido é transformado para string para ser armazenado na caixa de texto.
7. A transformação para string é necessária devido a propriedade Text, de um objeto da classe TextBox ser do tipo de dado string, e, sendo assim, não é possível, nesta propriedade, armazenar um valor numérico.

## FUNÇÃO

Como é do conhecimento da programação básica, a função é uma rotina que retorna valor. Sempre retorna valor. Sendo assim, quando uma função é chamada do evento clique, esta retorna valor para este evento que chamou a função, ou outra rotina (que chamou a função).

Os programas implementados no tópico anterior para calcular o quadrado de um número e, o segundo, para efetuar a soma de quatro valores numéricos, exibindo ou mostrando o resultado no próprio evento clique, serão aqui implementados novamente, porém com o conceito de função. Lembre-se que função retorna valor.

### 1. Exercício proposto:

Implementar um programa que chama uma função, denominada de CalculaQuadrado, e esta função recebe como parâmetro o valor numérico contido em uma caixa de texto, no momento da chamada desta função, com o objetivo de calcular o quadrado deste valor numérico passado por parâmetro. A função deverá ser chamada no evento clique do botão btQuadrado (sendo assim, o valor retorna para este evento clique, o qual a função é chamada).

O formulário para implementar o programa é o mesmo do tópico anterior, programa número 1, que tem finalidade de calcular o quadrado e exibir o resultado no próprio evento clique.

O código para efetuar o programa anterior com função é:

```

1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Windows.Forms;
9
10 namespace ProcessaQuadradoFuncao
11 {
12     public partial class frmCalculaQuadrado : Form
13     {
14         public frmCalculaQuadrado()
15         {
16             InitializeComponent();
17         }
18
19         private void btQuadrado_Click(object sender, EventArgs e)
20         {
21             txtQuadrado.Text = Convert.ToString(CalculaQuadrado(
22                                     Convert.ToDecimal (txtValor.Text)));
23         }
24
25         private decimal CalculaQuadrado(decimal Valor)
26         {
27             return Valor * Valor;
28         }
29     }
30 }

```

Análise o código anterior, principalmente a chamada da função, `CalculaQuadrado`, e a sua implementação logo a seguir. Observe que a função é implementada retornando um valor do tipo de dado decimal e, no evento clique, quando o valor é retornado pela função, este valor retornado, é transformado em string. Lembre-se que a propriedade `Text` de um objeto da classe `TextBox` é do tipo string, e, sendo assim, não permite armazenar um valor do tipo de dado decimal.

A passagem do parâmetro é do tipo decimal, portanto, a função o recebe deste mesmo tipo. Ao calcular o quadrado, o valor retornado é do tipo de dado que o parâmetro `Valor`, pois, ao multiplicar dois valores numéricos do mesmo tipo de dados, eles terão o mesmo tipo de dados. No entanto, se um dos tipos aceita ponto decimal e o outro valor numérico é inteiro, então o resultado também aceita ponto decimal.

O segundo exemplo será agora implementado com o conceito de função, considerando que é a mesma interface, o mesmo formulário com mesmos objetos nele contidos. Releia o exercício e analise a implementação:

## 2. Exercício proposto:

Implementar um programa que chama uma função, denominada de `CalculaSoma`, e obtém quatro valores numéricos de objetos da classe `TextBox`, que deverão ser passados para a função como parâmetros no momento da sua chamada. As caixas de texto são denominadas de `txtValor1`, `txtValor2`,



txtValor3 e txtValor4, e calcular a soma dos conteúdos numéricos destas caixas de textos. Exibir o resultado em um outro objeto denominado de txtSoma, também pertencente à classe TextBox. A chamada da função deverá ocorrer no evento clique do botão btSoma.

```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Windows.Forms;
9
10 namespace ProcessaSomaFuncao
11 {
12     public partial class Form1 : Form
13     {
14         public Form1()
15         {
16             InitializeComponent();
17         }
18
19         private void btSoma_Click(object sender, EventArgs e)
20         {
21             txtSoma.Text = CalculaSoma(Convert.ToDecimal(txtValor1.Text),
22                                     Convert.ToDecimal(txtValor2.Text),
23                                     Convert.ToDecimal(txtValor3.Text),
24                                     Convert.ToDecimal(txtValor4.Text));
25         }
26
27         private string CalculaSoma(decimal V1, decimal V2,
28                                   decimal V3, decimal V4)
29         {
30             return Convert.ToString(V1 + V2 + V3 + V4);
31         }
32     }
33 }
34 }
```

É importante que o leitor analise o código anterior e observe como os conteúdos das caixas de texto estão sendo transformados ou convertidos para numérico, decimal, e que o tipo de valor retornado pela função é string e, sendo assim, a soma efetuada na função é convertida para string para que o comando return possa retornar o valor da soma com o mesmo tipo de dados da declaração da função, que é string.

## PROCEDIMENTO

Como foi explanado nos parágrafos anteriores, em um procedimento o resultado do processamento pode ser apresentado ou mostrado no próprio procedimento ou, por parâmetro, exibido ou mostrado no evento clique que chamou o procedimento, ou em uma outra rotina qualquer, inclusive evento, que chamou o procedimento.

O próximo parágrafo aborda o processamento com procedimento exibindo o resultado do cálculo no próprio procedimento.

### Exibindo o Resultado no Próprio Procedimento

No intuito de melhor entender o conceito deste tópico, o próximo item contém o exercício de número um, cálculo do quadrado de um número. Compare-o com a solução obtida por função e demais metodologias. Segue o enunciado:

#### 1. Exercício proposto:

Implementar um programa que chama um procedimento, denominado de `CalculaExibeQuadrado`, e este procedimento recebe como parâmetro o valor numérico contido em uma caixa de texto, no momento da chamada deste procedimento, com o objetivo de calcular o quadrado deste valor numérico passado por parâmetro. O procedimento deverá ser chamado do evento clique de um botão, objeto da classe `btButton`, e o resultado do processamento deverá ser mostrado ou apresentado ao usuário pelo próprio procedimento. O objeto que recebe o valor processado para ser mostrado é da classe `TextBox`, denominado de `txtQuadrado`.

O código que contém a solução deste exercício encontra-se a seguir:

```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Windows.Forms;
9 namespace ProcessaQuadradoExibeProc
10 {
11     public partial class frmCalculaQuadradoExibeProc : Form
12     {
13         public frmCalculaQuadradoExibeProc()
14         {
15             InitializeComponent();
16         }
17         private void btQuadrado_Click(object sender, EventArgs e)
18         {
19             CalculaExibeQuadrado(Convert.ToDecimal(txtValor.Text));
20         }
21         private void CalculaExibeQuadrado(decimal V)
22         {
23             txtQuadrado.Text = Convert.ToString(V * V);
24         }
25     }
26 }
```

O código anterior, que chama do evento clique btQuadrado (linha 20) um procedimento (veja na linha 13), cujo tipo de dado do procedimento é void, veja nesta mesma linha 13. O cálculo do quadrado é efetuado multiplicando o conteúdo do parâmetro (V) por ele mesmo, e o resultado é convertido para string, para que possa ser armazenado na propriedade Text, que é do tipo de dado string.

O segundo exercício requer, como já visto, efetuar o cálculo da soma de quatro valores obtidos de objetos da classe TextBox. O enunciado deste exercício encontra-se a seguir:

## 2. Exercício proposto:

Implementar um programa que chama um procedimento, denominado de CalculaExibeSoma, e este procedimento recebe como parâmetros os valores numéricos contido em quatro caixas de texto, denominadas de txtValor1, txtValor2, txtValor3 e txtValor4. O procedimento deverá receber os parâmetros e calcular e exibir a soma destes parâmetros. O procedimento deverá ser chamado do evento clique de um botão, objeto da classe btButton, e o resultado do processamento deverá ser mostrado ou apresentado ao usuário pelo próprio procedimento. O objeto que recebe o valor processado para ser mostrado é da classe TextBox, denominado de txtSoma.

O código que implementa este exercício encontra-se a seguir:

```
1  using System.ComponentModel;
2  using System.Data;
3  using System.Text;
4  using System.Windows.Forms;
5
6  namespace ProcessaExibeProc
7  {
8      public partial class frmcalculaSomaExibeProc : Form
9      {
10         public frmcalculaSomaExibeProc()
11         {
12             InitializeComponent();
13         }
14
15         private void btSoma_Click(object sender, EventArgs e)
16         {
17             CalculaSoma(Convert.ToDecimal(txtValor1.Text),
18                         Convert.ToDecimal(txtValor2.Text),
19                         Convert.ToDecimal(txtValor3.Text),
20                         Convert.ToDecimal(txtValor4.Text));
21         }
22
23         private void CalculaSoma(decimal V1, decimal V2,
24                                 decimal V3, decimal V4)
25         {
26             txtSoma.Text = Convert.ToString(V1 + V2 + V3 + V4);
27         }
28     }
29 }
```

Os programas a seguir deverão ser implementados com os procedimentos exibindo o resultado no próprio procedimento. Para implementar o procedimento exibindo o resultado no próprio procedimento, basta passar como parâmetros os conteúdos das caixas de textos necessárias para efetuar o cálculo e exibir o resultado no próprio procedimento.

1- Implementar um programa que chama um procedimento do evento clique de um objeto da classe Button denominado de btQuadrado, para calcular e exibir o quadrado de um valor numérico lido de um TextBox neste evento clique e que deve ser passado para o procedimento no momento de sua chamada. O nome do procedimento é ExibeQuadrado. A propriedade Text do objeto da classe Button é &Processar.

2- Desenvolver um programa que chama uma função, denominada de Quadrado, no evento Click de um objeto da classe Button, denominado de btQuadrado. Passar para esta função um valor numérico lido de um objeto da classe TextBox, para que a mesma retorne o quadrado deste valor que deverá ser exibido no evento Click.

3- Fazer um programa que chama um procedimento, denominado de ConcatenaExibeStrings, para concatenar três strings lidas de 3 objetos da classe TextBox. Os conteúdos dos três objetos da classe TextBox deverão ser passados para o procedimento no momento de sua chamada. O procedimento deverá ser chamado do evento clique de um objeto da classe Button que tem a propriedade name igual a btConcatena e exibir o resultado no próprio procedimento.

4- Implementar um programa que chama uma função, denominada de ConcatenaStrings, para concatenar três strings lidas de 3 objetos da classe TextBox e que deverão ser passados para a função no momento de sua chamada. A função deverá ser chamada do evento clique de um objeto da classe Button que tem a propriedade name igual a btConcatena. O nome da função é ConcatenaStrings.

5- Faça uma função para retornar o valor da soma de dois valores que deverão ser passados para a função no momento de sua chamada. A função deverá ser chamada no evento click de um objeto da classe Button denominado de btProcessar. Os valores são obtidos de objetos da classe TextBox e exibido em um objeto da classe Label, denominado de lblSoma, pelo mesmo evento click.

6 – Desenvolver o programa anterior com um procedimento, exibindo o resultado no próprio procedimento com o nome de CalculaExibeSoma.

7- Faça um programa para obter três valores numéricos de quatro objetos da classe TextBox, no evento clique de um objeto da classe Button, denominado de btMultiplica. Chamar um procedimento, denominado de CalculaExibeProduto, para efetuar a multiplicação dos três valores que deverão ser passados para o procedimento no momento de sua chamada e exibir o resultado no próprio procedimento.

8- Implementar o programa anterior com uma função de nome CalculaProduto.

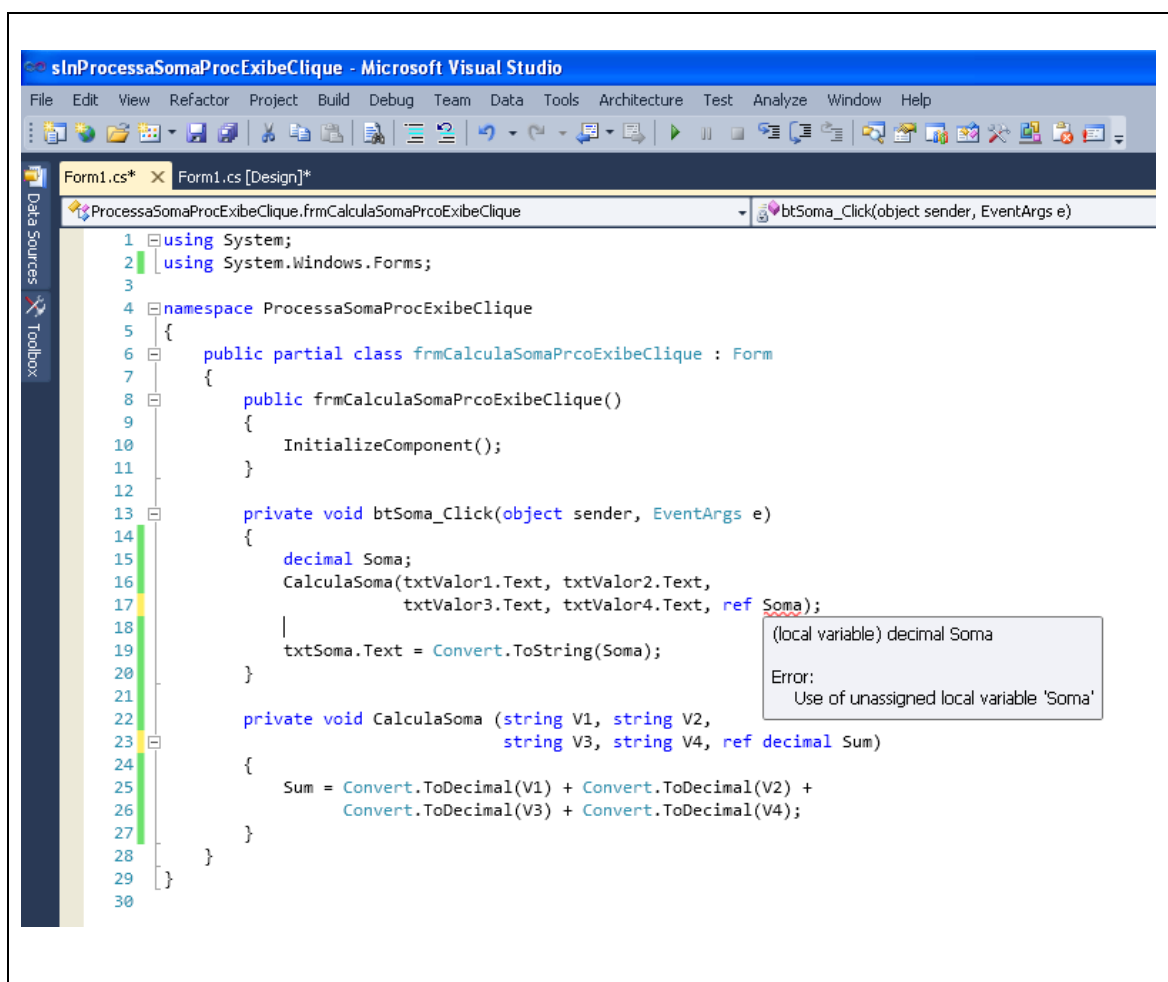
#### Exibindo o Resultado na Rotina (ou evento) que Chama o Procedimento

Observe que este item é chamado um procedimento, no entanto, o procedimento não exibe o resultado do processamento. Este resultado é exibido ou mostrado no evento clique ou na rotina que chamou o procedimento.

## 1. Exercício proposto:

Implementar um programa que chama um procedimento e recebe por parâmetros os quatro valores numéricos de objetos da classe TextBox, denominados de txtValor1, txtValor2, txtValor3 e txtValor4. O procedimento calcula a soma dos conteúdos numéricos destas caixas de textos. Exibir o resultado em um outro objeto denominado de txtSoma, também pertencente à classe TextBox. A chamada do procedimento deverá ocorrer quando o usuário clicar no botão btSoma. Exibir o resultado no evento clique, btSoma, que chamou o procedimento.

O código que implementa este exercício encontra-se a seguir:



The screenshot shows the Microsoft Visual Studio IDE with a C# file named `Form1.cs` open. The code is for a class `frmCalculaSomaPrcoExibeClique` which inherits from `Form`. It contains a constructor, an `InitializeComponent()` call, and two event handler methods. The `btSoma_Click` method calls `CalculaSoma` with four text box values and a `ref decimal Soma` parameter. The `CalculaSoma` method calculates the sum of four decimal values and returns it via the `ref` parameter. A compilation error is shown in a tooltip: "(local variable) decimal Soma Error: Use of unassigned local variable 'Soma'". The error occurs because the `ref` parameter is used before being assigned a value.

```
1 using System;
2 using System.Windows.Forms;
3
4 namespace ProcessaSomaProcExibeClique
5 {
6     public partial class frmCalculaSomaPrcoExibeClique : Form
7     {
8         public frmCalculaSomaPrcoExibeClique()
9         {
10             InitializeComponent();
11         }
12
13         private void btSoma_Click(object sender, EventArgs e)
14         {
15             decimal Soma;
16             CalculaSoma(txtValor1.Text, txtValor2.Text,
17                 txtValor3.Text, txtValor4.Text, ref Soma);
18             txtSoma.Text = Convert.ToString(Soma);
19         }
20
21         private void CalculaSoma (string V1, string V2,
22             string V3, string V4, ref decimal Sum)
23         {
24             Sum = Convert.ToDecimal(V1) + Convert.ToDecimal(V2) +
25                 Convert.ToDecimal(V3) + Convert.ToDecimal(V4);
26         }
27     }
28 }
29
30
```

O código acima que tem a pretensão de implementar a soma com passagem de parâmetros, exibindo ou mostrando o resultado no evento clique do botão btSoma, apresenta um erro, erro este denominado de “Use of unassigned local variable ‘Soma’ ”. Traduzindo: “uso da variável local, ‘Soma’, não inicializada”.

No intuito de alcançar êxito, o objetivo desejado, altere a palavra reservada `ref` por `out`. Observe o resultado final após a compilação do programa. O Novo código, desta forma alterada, é como se segue:

```

1  using System;
2  using System.Windows.Forms;
3
4  namespace ProcessaSomaProcExibeClique
5  {
6      public partial class frmCalculaSomaPrcoExibeClique : Form
7      {
8          public frmCalculaSomaPrcoExibeClique()
9          {
10             InitializeComponent();
11         }
12
13         private void btSoma_Click(object sender, EventArgs e)
14         {
15             decimal Soma;
16             CalculaSoma(txtValor1.Text, txtValor2.Text,
17                         txtValor3.Text, txtValor4.Text, out Soma);
18
19             txtSoma.Text = Convert.ToString(Soma);
20         }
21
22         private void CalculaSoma (string V1, string V2,
23                                   string V3, string V4, out decimal Sum)
24         {
25             Sum = Convert.ToDecimal(V1) + Convert.ToDecimal(V2) +
26                  Convert.ToDecimal(V3) + Convert.ToDecimal(V4);
27         }
28     }
29 }

```

A palavra reservada **out** especifica que a passagem de parâmetro deve ser por intermédio da variável que se encontra precedida com out para que o valor da mesma possa retornar para a chamada do procedimento, através do parâmetro que a ela corresponde.

Com a palavra reservada **ref** também é feito a passagem de parâmetros e exibe o resultado no evento clique do mesmo jeito, no entanto, esta variável que é precedida pelo ref necessita de ser inicializada, ou zerada (que não deixa de ser uma inicialização).

O erro que é ilustrado na página anterior, “Use of unassigned local variable ‘Soma’ ”, é provocado por esta situação, em que a variável Soma é passada como ref e não é inicializada.

Veja que o próximo código usa o ref, e para que não ocorra o erro já especificado, é apenas suficiente a inicialização desta variável (pode ser com zero):

```

1  using System;
2  using System.Windows.Forms;
3
4  namespace ProcessaSomaProcExibeClique
5  {
6      public partial class frmCalculaSomaPrcoExibeClique : Form
7      {
8          public frmCalculaSomaPrcoExibeClique()
9          {
10             InitializeComponent();
11         }
12
13         private void btSoma_Click(object sender, EventArgs e)
14         {
15             decimal Soma = 0;
16             CalculaSoma(txtValor1.Text, txtValor2.Text,
17                 txtValor3.Text, txtValor4.Text, out Soma);
18
19             txtSoma.Text = Convert.ToString(ref Soma);
20         }
21
22         private void CalculaSoma (string V1, string V2,
23             string V3, string V4, ref decimal Sum)
24         {
25             Sum = Convert.ToDecimal(V1) + Convert.ToDecimal(V2) +
26                 Convert.ToDecimal(V3) + Convert.ToDecimal(V4);
27         }
28     }
29 }

```

Observe que a linha 15 tem a variável Soma inicializada com zero e a passagem de parâmetros passou a ser com a palavra reservada ref. O procedimento quando executado calcula a soma e o resultado é passado para o evento clique por intermédio da variável Soma e Sum, cuja passagem de parâmetro é com a palavra reservada ref. Mas lembre-se que com esta palavra reservada, ref, é necessário inicializar a variável que vai conter o valor a ser passado para o procedimento, antes de sua chamada.

## 2. Exercício proposto:

Implementar um programa que chama um procedimento, denominado de CalculaQuadrado, e este procedimento recebe como parâmetro o valor numérico contido em uma caixa de texto, no momento da chamada deste procedimento, com o objetivo de calcular o quadrado deste valor numérico passado por parâmetro. O procedimento deverá ser chamado do evento clique de um botão, objeto da classe btButton, denominado de btSoma, e o resultado do processamento deverá ser mostrado ou apresentado ao usuário pelo evento clique que chamou o procedimento. O objeto que recebe o valor processado para ser mostrado é da classe TextBox, denominado de txtSoma.

A seguir tem-se o código deste exercício implementado. Verifique a chamada do procedimento e o parâmetro que vai conter o resultado deverá ser precedido da palavra reservada ref ou out. Caso seja ref, é necessário que esta variável seja inicializada. Se a palavra reservada é out, tal procedimento, não é necessário.

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Windows.Forms;
9
10 namespace ProcessaQuadradoExibeNaoProc
11 {
12     public partial class frmCalculaQuadrado : Form
13     {
14         public frmCalculaQuadrado()
15         {
16             InitializeComponent();
17         }
18
19         private void btQuadrado_Click(object sender, EventArgs e)
20         {
21             decimal Quadrado;
22             CalculaQuadrado(Convert.ToDecimal(Convert.ToDecimal(txtValor.Text)),
23                             out Quadrado);
24             txtQuadrado.Text = Convert.ToString(Quadrado);
25         }
26
27         private void CalculaQuadrado(decimal V, out decimal Quad)
28         {
29             Quad = V * V;
30         }
31     }
32 }

```

## EXERCÍCIOS

Os programas a seguir deverão ser implementados exibindo o resultado no evento clique onde o procedimento é chamado. Para implementar o procedimento é necessário, além de passar como parâmetros os conteúdos das caixas de textos necessárias para efetuar o cálculo, acrescentar mais um parâmetro para conter o resultado do cálculo, e exibir seu conteúdo no evento clique, onde o procedimento deverá ser chamado. Esta variável que é acrescentada possui passagem de parâmetro por referência (ByRef), para que seu conteúdo possa ser alterado no procedimento e também na chamada do procedimento.

1- Implementar um programa que chama um procedimento do evento clique do objeto da classe Button denominado de btQuadrado para calcular e exibir o quadrado de um valor numérico lido de um TextBox neste evento clique e que deve ser passado para o procedimento no momento de sua chamada. O nome do procedimento é CalculaQuadrado. A propriedade Text do objeto da classe Button é &Processar.

2- Desenvolver um programa que chama uma função, denominada de CalculaQuadrado, no evento Click de um objeto da classe Button, denominado de btQuadrado. Passar para esta função um valor numérico lido de um objeto da classe Text Box, para que a mesma retorne o quadrado deste valor que deverá ser exibido no evento Click.



3- Fazer um programa que chama um procedimento, denominado de ConcatenaStrings, para concatenar três strings lidas de 3 objetos da classe TextBox. Os conteúdos dos três objetos da classe TextBox deverão ser passados para o procedimento no momento de sua chamada. O procedimento deverá ser chamado do evento clique de um objeto da classe Button que tem a propriedade name igual a btConcatena.

4- Implementar um programa que chama uma função, denominada de ConcatenaStrings, para concatenar três strings lidas de 3 objetos da classe TextBox e que deverão ser passados para a função no momento de sua chamada. A função deverá ser chamada do evento clique de um objeto da classe Button que tem a propriedade name igual a btConcatena.

5- Faça uma função para retornar o valor da soma de dois valores que deverão ser passados para a função no momento de sua chamada. A função deverá ser chamada no evento click de um objeto da classe Button denominado de btProcessar. Os valores são obtidos de objetos da classe TextBox e exibido em um objeto da classe Label, denominado de lblsoma, pelo mesmo evento click.

6 – Desenvolver o programa anterior com um procedimento.

7- Faça um programa para obter três valores numéricos de quatro objetos da classe TextBox', no evento clique de um objeto da classe Button, denominado de btMultiplica. Chamar um procedimento, denominado de CalculaProduto, para efetuar a multiplicação dos três valores que deverão ser passados para o procedimento no momento de sua chamada.

8- Implementar o programa anterior com uma função com o nome de calculaProduto.

9- Implementar um programa para calcular a divisão de dois valores obtidos de objetos da classe TextBox, denominados de txtValor1 e txtValor2, e chamar um procedimento para calcular e exibir o resultado da operação de divisão dos valores numéricos que deverão ser passados para o procedimento como parâmetro. O procedimento deverá ser chamado do evento clique de um objeto da classe Button, denominado de ptDividir. O resultado deverá ser apresentado ou exibido no próprio procedimento em um objeto da classe TextBox, denominado de txtDivisao, o qual deverá também ser passado como parâmetro para o procedimento.

10- Implementar o programa anterior exibindo o resultado no próprio procedimento em um objeto da classe Label, denominado de lblDivisao, o qual também deverá ser passado para o procedimento como parâmetro.

11- Implementar um programa para calcular a soma, subtração, multiplicação, divisão, concatenação e potência de dois valores obtidos de caixas de textos, objetos da classe TextBox, denominados de txtValor1 e txtValor2, e exibir em um outro objeto, denominado de lblResultado, pertencente à classe Label. A operação a ser calculada deverá ser selecionada previamente e, em um determinado instante, somente uma operação poderá ser escolhida. O cálculo deverá ser efetuado ao clicar no objeto pertencente à classe Button, denominado de btProcessar. Cada uma das operações deverá ser implementada com procedimento e o resultado exibido ou apresentado no evento clique onde o procedimento é chamado.

12- Desenvolver o programa anterior considerando que cada operação deverá ser implementada com uma função.

13- Implementar o programa de número 11 com um procedimento que deverá receber também como parâmetros os valores numéricos dos objetos da classe TextBox, denominados de txtValor1 e txtValor2, a operação a ser efetuada (se é soma, subtração, multiplicação, etc.) e objeto da classe Label, lblResultado, que deverá exibir o resultado. Portanto, dois parâmetros a mais que nas rotinas dos dois programas anteriores.

## A CLASSE RadioButton

Implementar um programa para calcular a soma, subtração, multiplicação, divisão, concatenação e potência de dois valores obtidos de caixas de textos, objetos da classe TextBox, denominados de txtValor1 e txtValor2, e exibir em um outro objeto, denominado de lblResultado, pertencente à classe Label. A operação a ser calculada deverá ser selecionada previamente e, em um determinado instante, somente uma operação poderá ser escolhida. O cálculo deverá ser efetuado ao clicar no objeto pertencente à classe Button, denominado de btProcessar.

Neste programa, o único objeto que ainda não foi incluído em uma implementação é aquele que permite uma única seleção de uma opção a cada instante. Este objeto é da classe System.Windows.Forms.RadioButton.

A propriedade Checked, deste objeto, tipo de dados Boolean, especifica a opção que está selecionada em determinado instante, obviamente, quando o seu conteúdo for true. Quando uma opção de um conjunto de RadioButton possui o conteúdo desta variável igual a true, então todas as demais opções possuem conteúdos iguais a false. Cada opção é um objeto da classe RadioButton.

Por exemplo: seja o conjunto formado pelas opções rbMasculino e rbFeminino, dois objetos da classe RadioButton. Se um deles possui a propriedade Checked com o conteúdo verdadeiro, o outro objeto, imediatamente, passa ter o conteúdo desta propriedade falso. Portanto, um conjunto de objetos da classe RadioButton possui somente um dos objetos com conteúdo da propriedade Checked com valor igual true em um determinado instante.

Estes objetos, RadioButton, deverão estar contidos em um outro, denominado de GroupBox, para formar e gerenciar um conjunto de opções. Por exemplo: se a interface possui dois conjuntos diferentes de opções, uma para sexo e a outra para estado civil, cada conjunto deverá se encontrar contido no seu GroupBox: um para sexo e o outro para o estado civil, de modo que ao clicar em uma opção de um, não interfira nas demais opções do outro conjunto.

O objeto da classe GroupBox é para conter objetos de outras classes e fazer um separação, ou grupo, ou conjunto lógico que possuem as mesmas características, de modo que, as opções de um conjunto não interfira no outro. Veja mais uma vez o exemplo do parágrafo anterior: estado civil (com as opções casado, solteiro, viúvo, etc.) constitui um conjunto. Sexo constitui um outro conjunto (masculino e feminino). Ao selecionar uma opção de um destes conjuntos, não há nenhuma interferência no outro conjunto, pois cada conjunto especifica uma determinada característica.

O enunciado acima deverá ser implementado da seguinte forma:

1. Exibir o resultado no próprio evento clique do botão, objeto da classe btButton, btProcessar.

2. Implementar e chamar uma função para cada operação (exibir o resultado no evento clique que chamou a função).
3. Implementar e chamar um procedimento e no próprio procedimento exibir o resultado.
4. Implementar e chamar um procedimento, mas exibir o resultado no evento clique que chamou o procedimento.

A interface que segue apresenta os objetos que deverão se encontrar no formulário e atende as necessidades de processamento de qualquer um dos quatro itens solicitados no enunciado da questão:

Formulário com os objetos para processar o cálculo das operações básicas

Criar um novo projeto, e adicionar os objetos não existentes, configurando as suas propriedades, de acordo com as tabelas que seguem:

#### Objeto Form1 da Classe System.Windows.Forms.Form

PROPRIEDADE	VALOR	DESCRIÇÃO
Name	frmOperacoes	Já descrito.
Text	Cálculo das operações básicas com objetos da classe RadioButton	Já descrito.
Ico	FLGBRAZL.ICO	Já descrito.
MaximizeBox	False	Já descrito.
FormBorderStyle	FixedSingle	Já descrito.
Font	Bold	Já descrito.
StartPosition	CenterScreen	Já descrito.

#### Objeto Button1 da Classe System.Windows.Forms.Button

PROPRIEDADE	VALOR	DESCRIÇÃO
-------------	-------	-----------

<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	btProcessar	Já descrito.
<b>Text</b>	&Processar	Já descrito.

**Objeto Label1 da Classe System.Windows.Forms.Label**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	Label1	Já descrito.
<b>Text</b>	Valor 1	Já descrito.

**Objeto Label2 da Classe System.Windows.Forms.Label**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	Label2	Já descrito.
<b>Text</b>	Valor 2	Já descrito.

**Objeto Label3 da Classe System.Windows.Forms.Label**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	lblResultado	Já descrito.
<b>Text</b>	Resultado	Já descrito.

**Objeto Label3 da Classe System.Windows.Forms.Label**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	Label3	Já descrito.
<b>Text</b>	Resultado	Já descrito.

**Objeto Text1 da Classe System.Windows.Forms.TextBox**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Name</b>	txtValor1	Já descrito.
<b>Text</b>		Já descrito.

**Objeto Text1 da Classe System.Windows.Forms.TextBox**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Name</b>	txtValor2	Já descrito.
<b>Text</b>		Já descrito.

**Objeto GroupBox1 da Classe System.Windows.Forms.GroupBox**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
---------------------	--------------	------------------

<b>Name</b>	GroupBox1	Já descrito.
<b>Text</b>	Operações	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Checked</b>	True	Propriedade do tipo Boolean que seleciona um objeto quando seu conteúdo for True.
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbSomar	Já descrito.
<b>Text</b>	&Somar	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbSubtrair	Já descrito.
<b>Text</b>	S&ubtrair	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbMultiplicar	Já descrito.
<b>Text</b>	&Multiplicar	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbDividir	Já descrito.
<b>Text</b>	&Dividir	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbConcatenar	Já descrito.
<b>Text</b>	&Concatenar	Já descrito.

**Objeto RadioButton1 da Classe System.Windows.Forms.RadioButton**

<b>.PROPRIEDADE</b>	<b>VALOR</b>	<b>DESCRIÇÃO</b>
<b>Font subopção Bold</b>	True	Já descrito.
<b>Name</b>	rbPotencia	Já descrito.
<b>Text</b>	&Potência	Já descrito.

O código para processar a operação do item um, processamento e apresentação do resultado no próprio evento clique do botão, objeto da classe `btButton`, `btProcessar`, cuja opção deverá ser selecionada, e o botão com o rótulo Processar ser clicado, é da seguinte forma:

1. Solução do item um (exibir o resultado no próprio evento clique do botão, objeto da classe `btButton`, `btProcessa`)

```
1 using System;
2 using System.Windows.Forms;
3
4 namespace CalculadoraRadioButtonExibeClique
5 {
6     public partial class Form1 : Form
7     {
8         public Form1()
9         {
10             InitializeComponent();
11         }
12
13         private void btProcessar_Click(object sender, EventArgs e)
14         {
15             if (rbSomar.Checked)
16                 txtResultado.Text = Convert.ToString(Convert.ToDecimal(txtValor1.Text) +
17                                                         Convert.ToDecimal(txtValor2.Text));
18             else if (rbSubtrair.Checked)
19                 txtResultado.Text = Convert.ToString(Convert.ToDecimal(txtValor1.Text) -
20                                                         Convert.ToDecimal(txtValor2.Text));
21             else if (rbMultiplicar.Checked)
22                 txtResultado.Text = Convert.ToString(Convert.ToDecimal(txtValor1.Text) *
23                                                         Convert.ToDecimal(txtValor2.Text));
24             else if (rbDividir.Checked)
25                 txtResultado.Text = Convert.ToString(Convert.ToDecimal(txtValor1.Text) /
26                                                         Convert.ToDecimal(txtValor2.Text));
27             else if (rbConcatenar.Checked)
28                 txtResultado.Text = txtValor1.Text + txtValor2.Text;
29
30             else if (rbPotencia.Checked)
31                 txtResultado.Text = Convert.ToString
32                     (Math.Pow((double)Convert.ToDecimal (txtValor1.Text),
33                               (double)Convert.ToDecimal (txtValor2.Text)));
34         }
35     }
36 }
```

Observe que o próprio evento clique do objeto da classe `Button`, exibe o resultado para cada operação. O comando `if ... else if` verifica qual objeto da classe `RadioButton` foi selecionado para que a operação possa ser efetuada.

Para calcular a potência de um valor elevado a um outro é necessário a classe `Math` que é estática. Sendo assim, ela não pode ser instanciada, e para aplicá-la no programa é suficiente seu nome seguido do método que também deve ser estático. Este conceito em detalhe será estudado nas próximas aulas. Para calcular a potência da variável denominada de `valor1` elevado ao conteúdo da variável `valor2`, as duas variáveis do tipo de dado `double`, tem-se:

`Math.Pow (Valor1, Valor2);`

2. Solução do item dois (implementar e chamar uma função para cada operação - exibir o resultado no evento clique que chamou a função)

```
using System;
using System.Windows.Forms;

namespace slnCalculadoraRadioButtoFuncao2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void btProcessar_Click(object sender, EventArgs e)
        {
            if (rbSomar.Checked)
                txtResultado.Text = CalculaSoma(txtValor1.Text, txtValor2.Text);
            else if (rbSubtrair.Checked)
                txtResultado.Text = CalculaSubtracao(txtValor1.Text, txtValor2.Text);
            else if (rbMultiplicar.Checked)
                txtResultado.Text = CalculaMultiplicacao(txtValor1.Text, txtValor2.Text);
            else if (rbDividir.Checked)
                txtResultado.Text = CalculaDivisao(txtValor1.Text, txtValor2.Text);
            else if (rbConcatenar.Checked)
                txtResultado.Text = CalculaConcatena(txtValor1.Text, txtValor2.Text);
            else if (rbPotencia.Checked)
                txtResultado.Text = CalculaPtencia(txtValor1.Text, txtValor2.Text);
        }

        private string CalculaSoma(string V1, string V2)
        {
            return Convert.ToString(Convert.ToDecimal(V1) + Convert.ToDecimal(V2));
        }

        private string CalculaSubtracao(string V1, string V2)
        {
            return Convert.ToString(Convert.ToDecimal(V1) - Convert.ToDecimal(V2));
        }

        private string CalculaMultiplicacao(string V1, string V2)
        {
            return Convert.ToString(Convert.ToDecimal(V1) * Convert.ToDecimal(V2));
        }

        private string CalculaDivisao(string V1, string V2)
        {
            return Convert.ToString(Convert.ToDecimal(V1) / Convert.ToDecimal(V2));
        }

        private string CalculaConcatena(string V1, string V2)
        {
            return V1 + V2;
        }
    }
}
```

```

    {
        return V1 + V2;
    }

    private string CalculaPtencia(string V1, string V2)
    {
        return Convert.ToString(Math.Pow(Convert.ToDouble(V1), Convert.ToDouble(V2)));
    }
}

```

Para o cálculo da potência é possível implementar, não transformando para double, como é o código da figura anterior, mas transformando para decimal e, em seguida, aplicando o conceito de cast ou moldagem da linguagem C, transformar para double em tempo de execução. Veja a figura a seguir com somente esta função:

```

private string CalculaPtencia(string V1, string V2)
{
    return Convert.ToString(Math.Pow((double)Convert.ToDecimal(V1),
                                     (double)Convert.ToDecimal(V2)));
}

```

Lembre-se que é possível também receber os parâmetros da função com o tipo de dados decimal e efetuar o cast ou moldagem para double. A função estática Pow, da classe Math, também estática, tem os tipos de dados double dos seus dois parâmetros, e não decimal. Se chamar a função passando parâmetro do tipo decimal, a função é implementada como a figura que segue:

```

private string CalculaPtencia(decimal V1, decimal V2)
{
    return Convert.ToString(Math.Pow((double) (V1),
                                     (double) (V2)));
}

```

Nesta última implementação é obrigatório a conversão do tipo de dados decimal para double (os tipos de dados dos parâmetros dos métodos Pow são do tipo double). Fato que não é necessário na implementação anterior a esta, tendo em vista que a transformação poderia ser diretamente feita para double, conforme a implementação no código original.

Através de uma estrutura else ... if, são chamadas as funções que processam cada uma das operações previamente selecionada.

A seguir é apresentada a mesma solução, do mesmo item dois, com a estrutura do comando switch.



```

using System;
using System.Windows.Forms;

namespace slnCalculadoraRadioButtoFuncao
{
    public partial class frmOperacoesComRadioButton : Form
    {
        public frmOperacoesComRadioButton()
        {
            InitializeComponent();
        }

        private void btProcessar_Click(object sender, EventArgs e)
        {
            char Op = '\0';
            Op = ObtemOperacao();
            switch (Op)
            {
                case '+':
                    txtResultado.Text = CalculaSoma(txtValor1.Text, txtValor2.Text);
                    break;
                case '-':
                    txtResultado.Text = CalculaSubtracao(txtValor1.Text, txtValor2.Text);
                    break;
                case '*':
                    txtResultado.Text = CalculaMultiplicacao(txtValor1.Text,
                                                                txtValor2.Text);
                    break;
                case '/':
                    txtResultado.Text = CalculaDivisao(txtValor1.Text, txtValor2.Text);
                    break;
                case 'c': // c de concatena
                    txtResultado.Text = CalculaConcatena(txtValor1.Text, txtValor2.Text);
                    break;
                default: // Se não é nenhuma operação, então será Potência
                    txtResultado.Text = CalculaPtencia(txtValor1.Text, txtValor2.Text);
                    break;
            }
        }

        private char ObtemOperacao()
        {
            if (rbSomar.Checked)
                return '+';
            else if (rbSubtrair.Checked)
                return '-';
            else if (rbMultiplicar.Checked)
                return '*';
            else if (rbDividir.Checked)
                return '/';
            else if (rbConcatenar.Checked)
                return 'c';
            else if (rbPotencia.Checked)
                return 'p';
            return '\0';
        }

        private string CalculaSoma(string V1, string V2)
        {
            return Convert.ToString(Convert.ToDecimal(V1) + Convert.ToDecimal(V2));
        }
    }
}

```

```

private string CalculaSubtracao(string V1, string V2)
{
    return Convert.ToString(Convert.ToDecimal(V1) - Convert.ToDecimal(V2));
}

private string CalculaMultiplicacao(string V1, string V2)
{
    return Convert.ToString(Convert.ToDecimal(V1) * Convert.ToDecimal(V2));
}

private string CalculaDivisao(string V1, string V2)
{
    return Convert.ToString(Convert.ToDecimal(V1) / Convert.ToDecimal(V2));
}

private string CalculaConcatena(string V1, string V2)
{
    return V1 + V2;
}

private string CalculaPtencia(string V1, string V2)
{
    return Convert.ToString (Math.Pow(Convert.ToDouble(V1),
                                     Convert.ToDouble(V2)) );
}
}
}

```

Observe que a potência é transformada para double, pois V1 e V2, os parâmetros da função, são do tipo de dados string. A insistência nestes tipos de dados é para chamar atenção para este conceito que é fundamental na linguagem C Sharp, como também nas demais linguagens.

Este código implementa a solução do item dois, chamando uma função, por meio do comando switch. A variável do comando é do tipo de dado char, e esta variável recebe um flag da operação a ser efetuada. O flag é constituído dos seguintes caracteres: a soma é o caracter '+', subtração caracter é '-', a multiplicação é '\*', divisão é '/', concatenação é 'c', e a potência, 'p'. Este último não é necessário, pois o default, caso nenhuma opção seja selecionada, ou nenhum dos case seja verdadeiro, é executado.

Para determinar qual a operação a ser efetuada foi implementada uma função que retorna o caracter, varável do tipo de dado char, com o flag correspondente da operação selecionada. É o valor retornado por esta função que se encontra no comando switch para processar a operação selecionada, conforme determina o parágrafo anterior.

### 3. Solução do item três (Implementar e chamar um procedimento e no próprio procedimento exibir o resultado)

Neste item, cada operação selecionada deve chamar um procedimento e no próprio procedimento exibir a operação selecionada. O código que se encontra na figura a seguir implementa a solução especificada:

```

using System;
using System.Windows.Forms;

namespace CalculadoraRadioButtonExibeProc
{
    public partial class frmCalculadoraExibeProc : Form
    {
        public frmCalculadoraExibeProc()
        {
            InitializeComponent();
        }

        private void btProcessar_Click(object sender, EventArgs e)
        {
            char Op = '\0';
            Op = ObtemOperacao();
            switch (Op)
            {
                case '+':
                    CalculaExibeSoma(txtValor1.Text, txtValor2.Text);
                    break;
                case '-':
                    CalculaExibeSubtracao(txtValor1.Text, txtValor2.Text);
                    break;
                case '*':
                    CalculaExibeMultiplicacao(txtValor1.Text, txtValor2.Text);
                    break;
                case '/':
                    CalculaExibeDivisao(txtValor1.Text, txtValor2.Text);
                    break;
                case 'c': // c de concatena
                    CalculaExibeConcatena(txtValor1.Text, txtValor2.Text);
                    break;
                default: // Se não é nenhuma operação, então será Potência
                    CalculaExibePtencia(txtValor1.Text, txtValor2.Text);
                    break;
            }
        }

        private char ObtemOperacao()
        {
            if (rbSomar.Checked)
                return '+';
            else if (rbSubtrair.Checked)
                return '-';
            else if (rbMultiplicar.Checked)
                return '*';
            else if (rbDividir.Checked)
                return '/';
            else if (rbConcatenar.Checked)
                return 'c';
            else if (rbPotencia.Checked)
                return 'p';
            return '\0';
        }

        private void CalculaExibeSoma(string V1, string V2)
        {
            txtResultado.Text = Convert.ToString(Convert.ToDecimal(V1) +
                                                    Convert.ToDecimal(V2));
        }
    }
}

```

```

private void CalculaExibeSubtracao(string V1, string V2)
{
    txtResultado.Text = Convert.ToString(Convert.ToDecimal(V1) -
                                          Convert.ToDecimal(V2));
}

private void CalculaExibeMultiplicacao(string V1, string V2)
{
    txtResultado.Text = Convert.ToString(Convert.ToDecimal(V1) *
                                          Convert.ToDecimal(V2));
}

private void CalculaExibeDivisao(string V1, string V2)
{
    txtResultado.Text = Convert.ToString(Convert.ToDecimal(V1) /
                                          Convert.ToDecimal(V2));
}

private void CalculaExibeConcatena(string V1, string V2)
{
    txtResultado.Text = V1 + V2;
}

private void CalculaExibePtencia(string V1, string V2)
{
    txtResultado.Text = Convert.ToString(Math.Pow(
        (double) Convert.ToDecimal(V1),
        (double) Convert.ToDecimal(V2)));
}
}

```

O código contido na figura anterior não é difícil de entendimento e, muito menos, existe alguma coisa não estudada no presente texto.

4. Solução do item quatro (Implementar e chamar um procedimento, mas exibir o resultado no evento clique que chamou o procedimento)

O código que segue contém este item:

```

using System;
using System.Windows.Forms;

namespace CalculadoraRadioButtonChamaProcExibeClick
{
    public partial class frmClculadoraRadioButtonChmaProcExibeClick : Form
    {
        public frmClculadoraRadioButtonChmaProcExibeClick()
        {
            InitializeComponent();
        }

        private void btProcessar_Click_1(object sender, EventArgs e)
        {

```

```

        decimal dcResultado;
        string strResultado;
        if (rbSomar.Checked) {
            CalculaSoma(txtValor1.Text, txtValor2.Text, out dcResultado);
            txtResultado.Text = Convert.ToString(dcResultado);
        }
        else if (rbSubtrair.Checked) {
            CalculaSubtracao(txtValor1.Text, txtValor2.Text,
                             out dcResultado);
            txtResultado.Text = Convert.ToString(dcResultado);
        }
        else if (rbMultiplicar.Checked) {
            CalculaMultiplicacao(txtValor1.Text, txtValor2.Text,
                                 out dcResultado);
            txtResultado.Text = Convert.ToString(dcResultado);
        }
        else if (rbDividir.Checked) {
            CalculaDivisao(txtValor1.Text, txtValor2.Text,
                           out dcResultado);
            txtResultado.Text = Convert.ToString(dcResultado);
        }
        else if (rbConcatenar.Checked) {
            CalculaConcatena(txtValor1.Text, txtValor2.Text,
                             out strResultado);
            txtResultado.Text = strResultado;
        }
        else if (rbPotencia.Checked) {
            CalculaPotencia(txtValor1.Text, txtValor2.Text,
                            out dcResultado);
            txtResultado.Text = Convert.ToString(dcResultado);
        }
    }

    private void CalculaSoma(string V1, string V2, out decimal dcR)
    {
        dcR = Convert.ToDecimal(V1) + Convert.ToDecimal(V2);
    }

    private void CalculaSubtracao(string V1, string V2, out decimal dcR)
    {
        dcR = Convert.ToDecimal(V1) - Convert.ToDecimal(V2);
    }

    private void CalculaMultiplicacao(string V1, string V2, out decimal dcR)
    {
        dcR = Convert.ToDecimal(V1) * Convert.ToDecimal(V2);
    }

    private void CalculaDivisao(string V1, string V2, out decimal dcR)
    {
        dcR = Convert.ToDecimal(V1) / Convert.ToDecimal(V2);
    }

    private void CalculaConcatena(string V1, string V2, out string strR)
    {
        strR = V1 + V2;
    }

    private void CalculaPotencia(string V1, string V2, out decimal dcR)
    {
        dcR = Convert.ToDecimal (Math.Pow((double)Convert.ToDecimal(V1),
                                           (double)Convert.ToDecimal(V2)) );
    }

```

```
}  
}  
}
```

Conforme o item anterior, este também não apresenta nenhuma novidade para a sua total compreensão, a não ser que alguma dúvida tenha ficado nos exercícios e exemplos anteriores, fato motivador para tal dúvida sejam tiradas e eliminadas.

Uma única dúvida que poderia surgir, dentro deste exemplo anterior, é o símbolo ‘{’ que inicia um bloco. Este símbolo inicia na mesma linha logo após a condição do comando if, após um espaço em branco, para melhor visualização, no próprio evento click, onde os procedimentos são chamados. Ele finaliza, com o símbolo ‘}’ na linha seguinte ao último comando do if, na mesma coluna que a cláusula else. A figura que segue contém um destes blocos:

```
else if (rbSubtrair.Checked) {  
    CalculaSubtracao(txtValor1.Text, txtValor2.Text,  
                    out dcResultado);  
  
    txtResultado.Text = Convert.ToString(dcResultado);  
}
```

## PARÂMETRO DO TIPO DE DADOS CONTROLE

É possível também implementar uma rotina (função ou procedimento) passando como parâmetro a própria caixa de texto que vai conter e exibir o resultado. No intuito de melhor entender e estender estes conceitos, uma nova especificação se faz necessário, conforme o enunciado que segue que determina a chamada da rotina e o resultado pode ser apresentado em um objeto da classe Label ou TextBox.

Como a rotina não sabe qual dos objetos, de qual das classes é passada como parâmetro, então é necessário uma verificação prévia para que se possa processar o objeto da classe correta. Vejamos primeiro o enunciado do programa a ser implementado:

Implementar um programa que chama um procedimento, denominado de CalculaQuadrado, para calcular o quadrado de um valor numérico obtido de um objeto da classe TextBox, denominado de txtValor, e passado por parâmetro para este procedimento o conteúdo desta caixa de texto, txtValor. O próprio procedimento deverá exibir o resultado em um objeto da classe TextBox ou Label (denominados de txtQuadrado ou lblQuadrado, respectivamente), de acordo com o que for passado por parâmetro para o procedimento. O procedimento deverá ser chamado do evento clique de um objeto da classe Button, denominado de btQuadrado.

Observe que o enunciado especifica que o procedimento deverá receber por parâmetro o objeto que vai conter o resultado e, este objeto, poderá ser da classe TextBox ou da classe Label. A figura a seguir contém o código deste procedimento e a sua chamada no evento clique de um objeto da classe Button.

Primeiramente, analise os objetos que deverão estar contidos no formulário, que o enunciado do programa especifica: três objetos no formulário. Objeto da classe TextBox, txtValor, um objeto da classe Label ou da classe TextBox (um ou outro, não ambos), denominados, respectivamente, de lblQuadrado ou txtQuadrado, e um objeto da classe Button, btQuadrado.

```
using System;
using System.Windows.Forms;

namespace CalculaQuadradoPorControleTextBox
{
    public partial class frmCalculaQuadradoPorControleTextBox : Form
    {
        public frmCalculaQuadradoPorControleTextBox()
        {
            InitializeComponent();
        }

        private void btQuadrado_Click(object sender, EventArgs e)
        {
            CalculaQuadrado(Convert.ToDecimal(txtValor.Text), txtQuadrado);
        }

        private void CalculaQuadrado(decimal Val, Object Controle)
        {
            if (Controle is TextBox)
            {
                TextBox oCaixaTexto;
                oCaixaTexto = (System.Windows.Forms.TextBox) Controle;
                oCaixaTexto.Text = Convert.ToString(Val * Val);
            }
            else if (Controle is Label)
            {
                Label oRotulo;
                oRotulo = (System.Windows.Forms.Label) Controle;
                oRotulo.Text = Convert.ToString(Val * Val); ;
            }
            else
            {
            }
        }
    }
}
```

Esta metodologia é importante quando se deseja que o procedimento ou função processe um cálculo e o seu resultado seja exibido sempre em um dos dois objetos, da classe TextBox ou da classe Label, não importando o nome do objeto passado por parâmetro ou qual das duas classes ele pertence, TextBox ou label.

Um detalhe também muito importante na implementação desta rotina é o fato de que a passagem de parâmetro de um objeto é sempre por referência.

## EXERCÍCIO

1. Implementar o mesmo programa do enunciado com um procedimento passando parâmetro de um objeto da classe Label para conter o resultado.
2. Implementar o programa do enunciado anterior com uma função que retorna o tipo de dado boolean (bool), de modo que se a função retornar false, então houve erro, e true, em caso contrário. Exibir o resultado em um objeto da classe Label na própria função.
3. Implementar o programa do enunciado anterior com uma função que retorna o tipo de dado boolean (bool), de modo que se a função retornar false, então houve erro, e true, em caso contrário. Exibir o resultado em um objeto da classe TextBox na própria função.

A seguir é implementada a solução do exercício de número 2 (dois) da última lista proposta, de modo que o aluno leitor tenha base suficiente para implementar os demais:

```
using System;
using System.Windows.Forms;

namespace slnCalculaQuadradoPorControleLabel
{
    public partial class frmCalculaQuadradoPorControleLabel : Form
    {
        public frmCalculaQuadradoPorControleLabel()
        {
            InitializeComponent();
        }

        private void btQuadrado_Click(object sender, EventArgs e)
        {
            if (!CalculaQuadrado(Convert.ToDecimal(txtValor.Text),
                                lblQuadrado) )
            {
                MessageBox.Show ("Parâmetro não esperado !",
                                "Erro de parâmetro");
            }
        }

        private bool CalculaQuadrado(decimal Val, Object Controle)
        {
            if (Controle is TextBox)
            {
                TextBox oCaixaTexto;
                oCaixaTexto = (System.Windows.Forms.TextBox)Controle;
                oCaixaTexto.Text = Convert.ToString(Val * Val);
                return true;
            }
            else if (Controle is Label)
            {
                Label oRotulo;
                oRotulo = (System.Windows.Forms.Label)Controle;
                oRotulo.Text = Convert.ToString(Val * Val);
                return true;
            }
        }
    }
}
```



```
        else  
            return false;  
    }  
}  
}
```