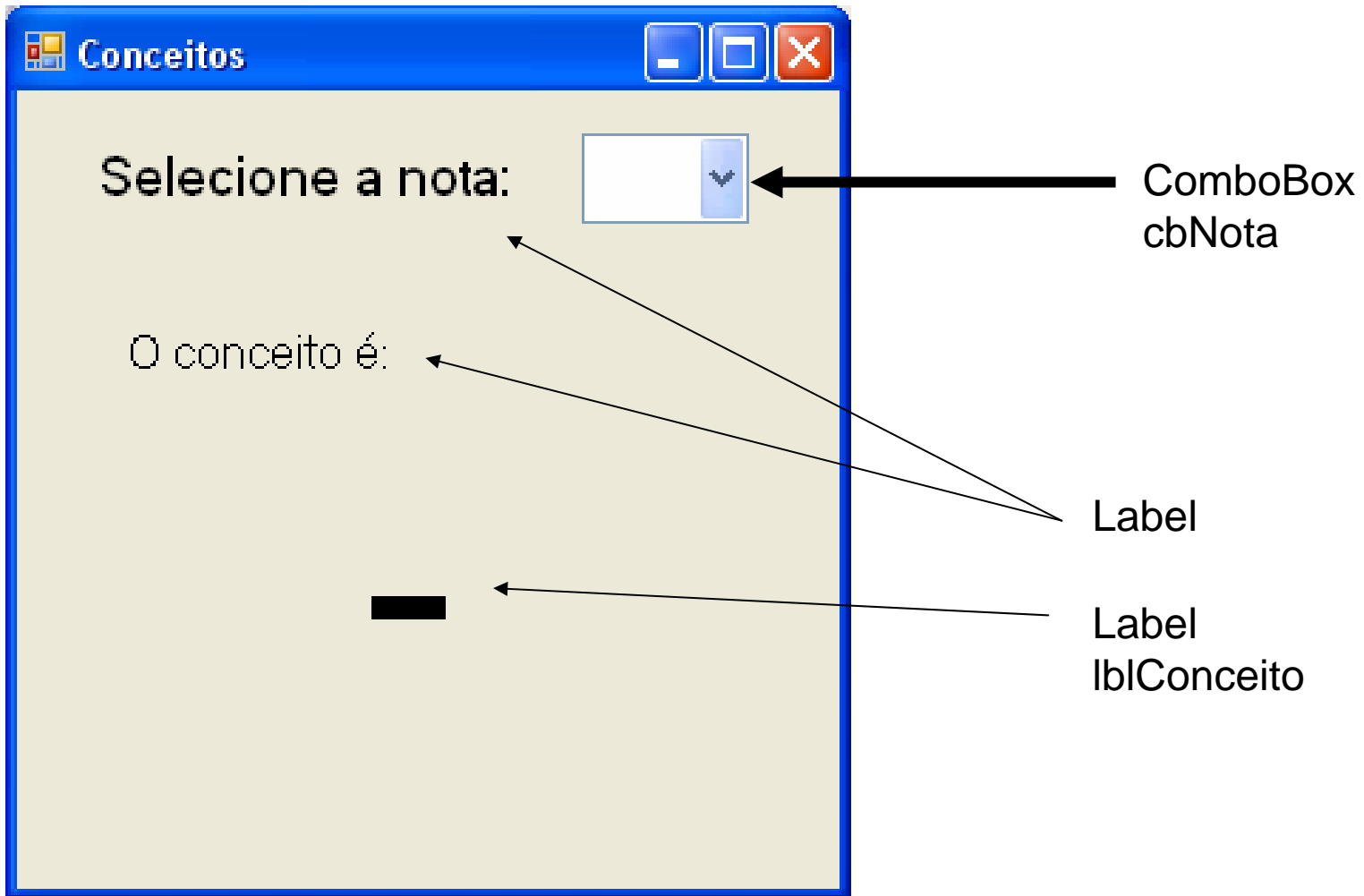


# Laboratório de Programação I

## Estruturas de Controle: Parte 2

Fabricio Breve

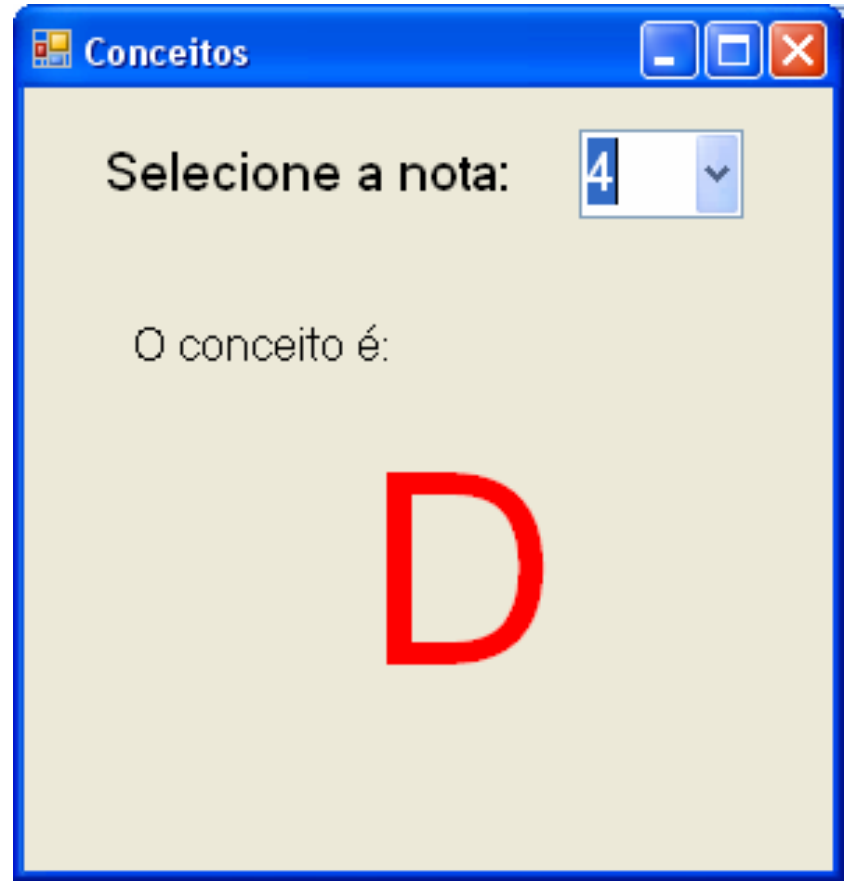
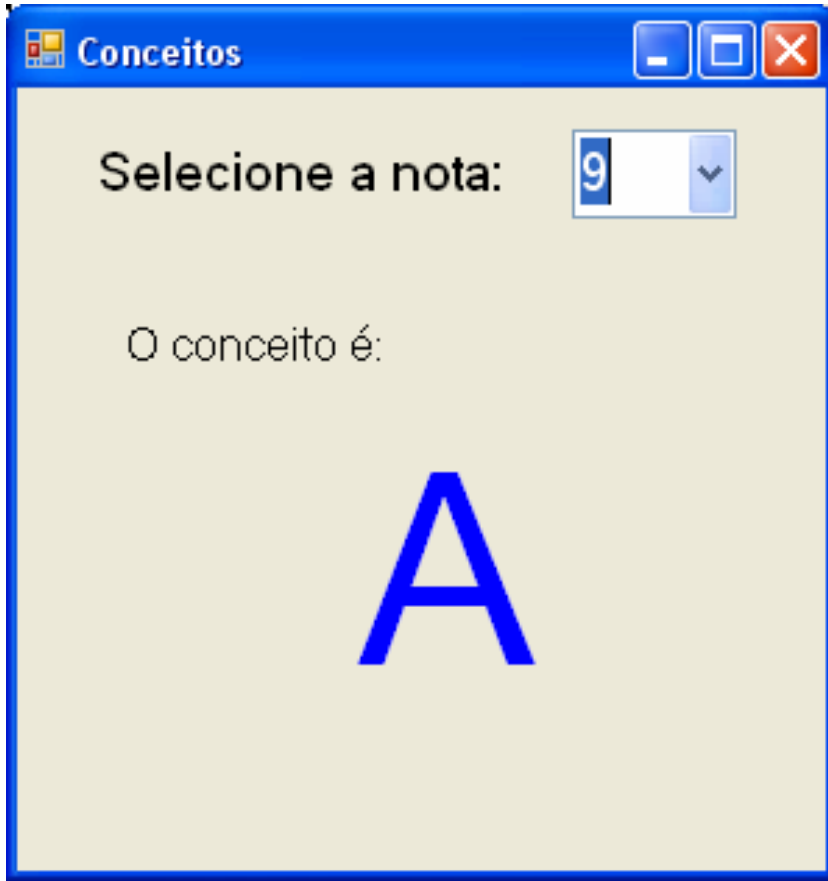
# Select Case



# Código do evento **SelectedIndexChanged** do componente **ComboBox**

```
Public Class formConceito
    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
cbNota.SelectedIndexChanged
        Select Case cbNota.SelectedItem
            Case Is >= 9
                lblConceito.Text = "A"
            Case Is >= 7
                lblConceito.Text = "B"
            Case Is >= 5
                lblConceito.Text = "C"
            Case Is >= 3
                lblConceito.Text = "D"
            Case Else
                lblConceito.Text = "E"
        End Select
    End Sub
End Class
```

# Adicione cores para os conceitos

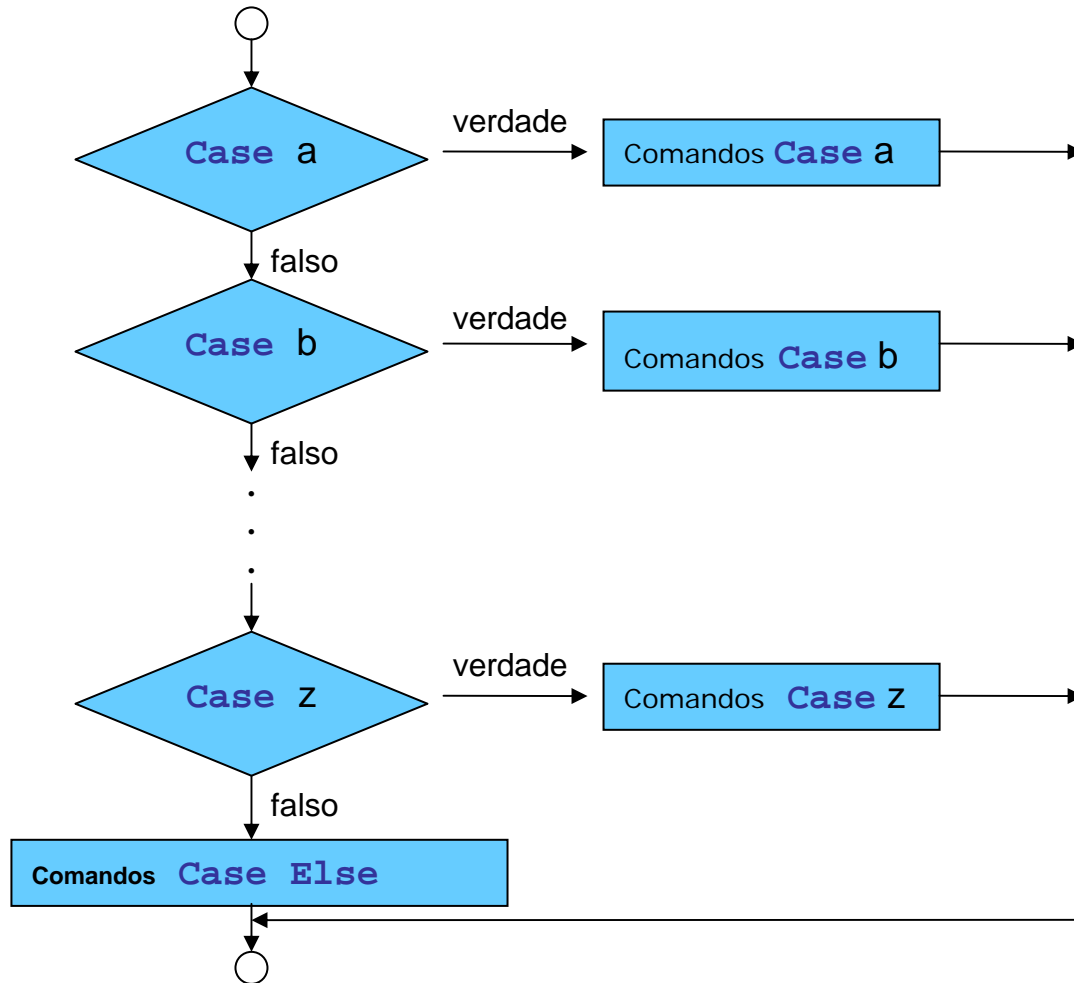


Dica: utilize a propriedade **ForeColor** do componente **Label**

# Select Case

- Estrutura de Seleção Múltipla
  - Testa cada expressão separadamente para cada valor que a expressão pode assumir
  - Palavra-chave: **Select Case**
    - Seguida pela expressão de controle
      - Comparada sequencialmente com cada caso
      - O código dentro do case é executado se uma igualdade for encontrada
      - O controle do programa prossegue para a primeira instrução após a estrutura
    - Palavra-chave: **Case**
      - Especifica cada valor a ser testado
      - Seguido por um código para executar se o teste for verdadeiro
      - **Case Else**
        - » Opcional
        - » Executa se nenhuma igualdade for encontrada
        - » Precisa ser o último caso da seqüência

# Fluxograma select Case



# Criando os itens do ComboBox dinamicamente

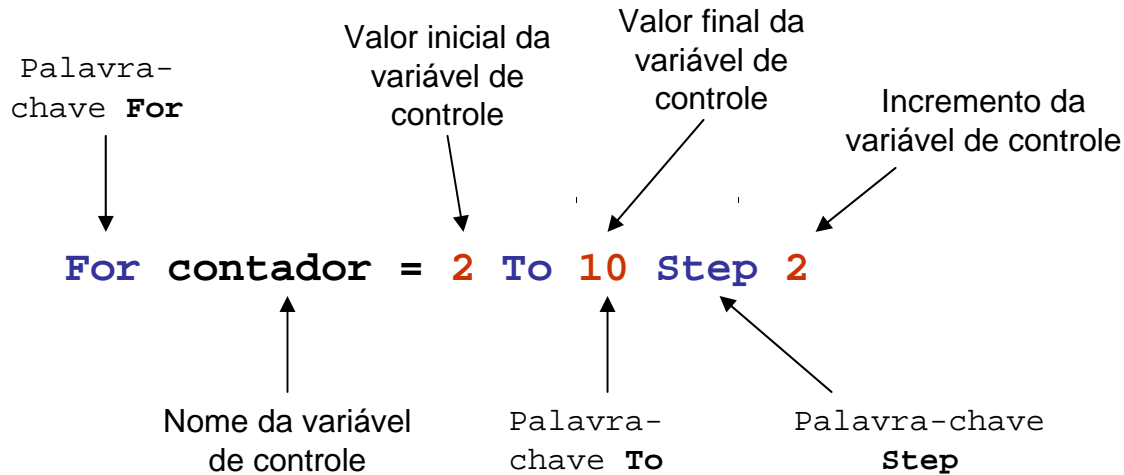
- Utilize o evento Load do Formulário

```
Private Sub formConceito_Load(ByVal sender As System.Object, ByVal  
e As System.EventArgs) Handles MyBase.Load  
    Dim nota As Integer  
    For nota = 0 To 10  
        cbNota.Items.Add(nota)  
    Next  
End Sub
```

# Estrutura de repetição: **For/Next**

- **For/Next** : repetição controlada por contador
  - O cabeçalho da estrutura inicializa a variável de controle e especifica um valor final e o incremento
    - **For** (palavra-chave que especifica o início da estrutura)
      - Seguido da inicialização da variável de controle
    - **To** (palavra-chave que especifica o valor final)
    - **Step** (palavra-chave que especifica o incremento)
      - Opcional
      - Incremento padrão 1 se omitido
      - Pode ser positivo ou negativo
    - **Next** (palavra-chave que marca o final da estrutura)
  - Executa enquanto a variável de controle for maior (ou igual) ao valor final

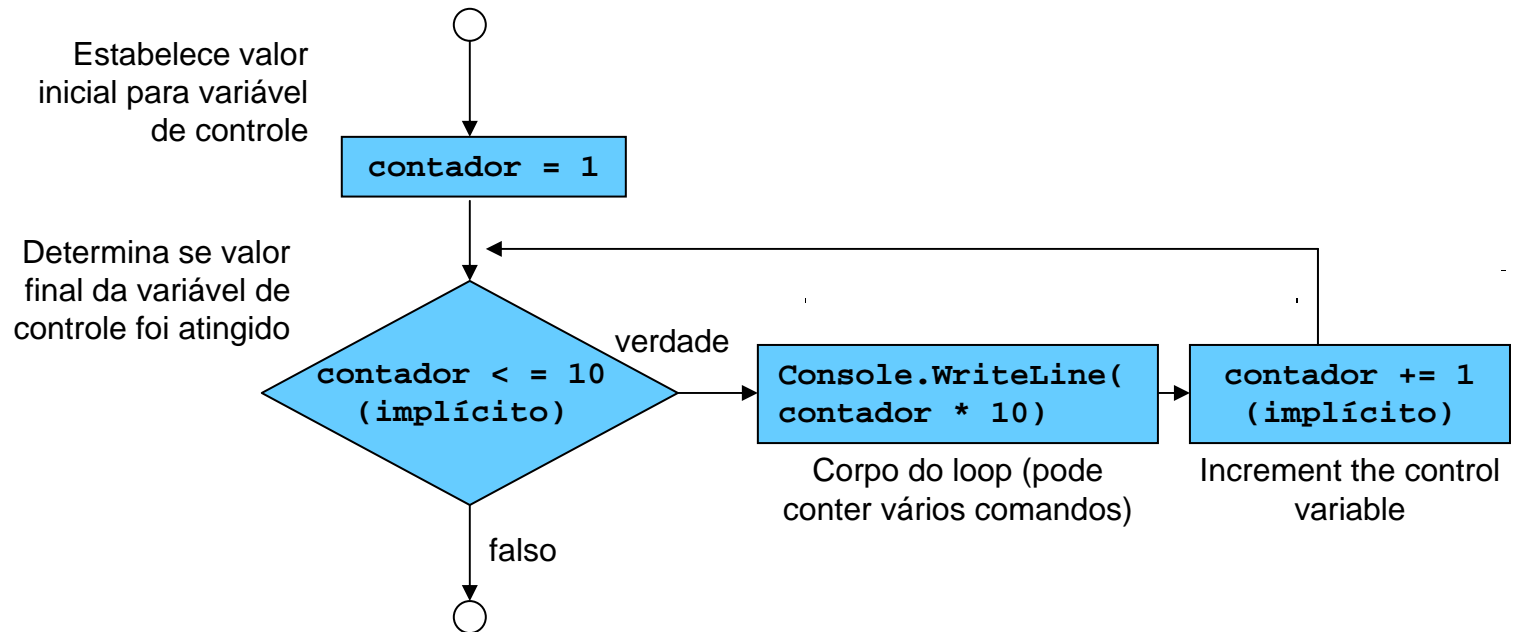
# Estrutura de repetição **For/Next**



# Exemplos usando a estrutura **For/Next**

- Exemplos
  - Variar a variável de controle de 1 a 100 em incrementos de 1
    - `For i = 1 To 100`
    - `For i = 1 To 100 Step 1`
  - Variar a variável de controle de 100 a 1 em decrementos de 1
    - `For i = 100 To 1 Step -1`
  - Variar a variável de controle de 7 a 77 em incrementos de 7
    - `For i = 7 To 77 Step 7`
  - Variar a variável de controle de 20 a 2 em decrementos de -2
    - `For i = 20 To 2 Step -2`

# Fluxograma For/Next

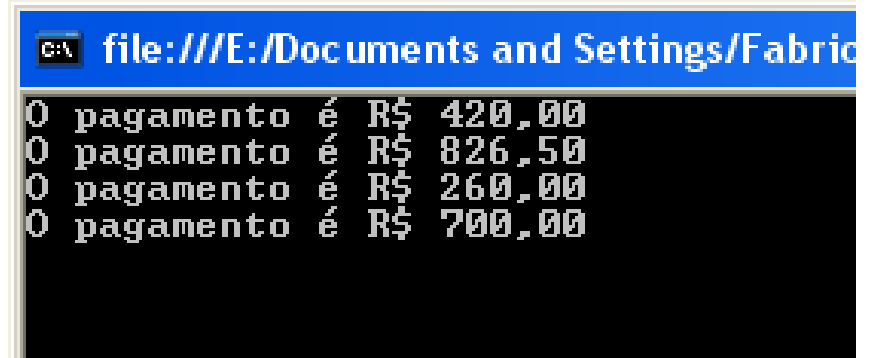


# Procedimentos `sub`

- Definição de um procedimento `sub`
  - `sub` nome-do-procedimento(lista-de-parâmetros)
  - declarações e comandos
  - `End Sub`
- Cabeçalho do procedimento
  - A primeira linha é conhecida como cabeçalho do procedimento
- Nome-do-Procedimento
  - Vem logo após a palavra-chave `sub`
  - Pode ser qualquer identificador válido
  - É usado para chamar este procedimento `sub` dentro do programa
- Corpo do procedimento
  - As declarações e comandos na definição do procedimento formam o corpo do procedimento

# Exemplo sub

```
Module modPagamento
  Sub Main()
    ' chama o procedimento ImprimirPagamento 4 vezes
    ImprimirPagamento(40, 10.5)
    ImprimirPagamento(38, 21.75)
    ImprimirPagamento(20, 13)
    ImprimirPagamento(50, 14)
    Console.ReadLine() ' impede a janela de fechar
  End Sub ' Main
  ' imprime o valor do pagamento na janela de comando
  Sub ImprimirPagamento(ByVal horas As Double, ByVal salariohora As Decimal)
    ' pagamento = horas * salariohora
    Console.WriteLine("O pagamento é {0:C}", horas * salariohora)
  End Sub ' ImprimirPagamento
End Module ' modPagamento
```



```
file:///E:/Documents and Settings/Fabric...
0 pagamento é R$ 420,00
0 pagamento é R$ 826,50
0 pagamento é R$ 260,00
0 pagamento é R$ 700,00
```

# Procedimentos **F**unction

- Similar aos procedimentos **s**ub
- Uma diferença importante
  - Procedimentos **F**unction retornam um valor para quem fez a chamada

# Procedimentos **Function**

- Formato de definição de um procedimento **Function**  
**Function** nome-do-procedimento(lista-de-param) **As** tipo-retorno  
declarações e comandos  
**End Function**
- Tipo-Retorno
  - Indica o tipo de dado do resultado retornado pela **Function** para seu chamador
- Expressão **Return**
  - Pode ocorrer em qualquer lugar de uma **Function**
  - Retorna exatamente um valor
  - O controle retorna imediatamente para o ponto no qual o procedimento foi invocado

# Exemplo de Function

```
Module modRaizQuadrada
    Sub Main()
        Dim i As Integer ' contador
        Console.WriteLine("Número" & vbTab & "Raíz" & vbCrLf)
        ' eleva números de 1 a 10 ao quadrado
        For i = 1 To 10
            Console.WriteLine(i & vbTab & Raiz(i))
        Next
        Console.ReadLine() ' Segura a janela aberta
    End Sub ' Main
    ' Função Raiz é executada
    ' somente quando a função é explicitamente chamada.
    Function Raiz(ByVal y As Integer) As Integer
        y = y ^ 2
        Return y 'Retorna y para a função que chamou Raiz()
    End Function ' Raiz
End Module ' modRaizQuadrada
```

# Saída do Exemplo de `Function`

```
C:\ file:///E:/Documents and Settings/Fabricio/Configurações locais/Dados de aplicativos/Temp...  
Número  Raíz  
1        1  
2        4  
3        9  
4       16  
5       25  
6       36  
7       49  
8       64  
9       81  
10      100  
-
```

# Tipos de Variáveis

Tipo	Tamanho em bits	Valores
Boolean	16	True ou False
Char	16	Um Caracter Unicode
Byte	8	0 a 255
Date	64	1 de Janeiro de 0001 a 31 de Dezembro de 9999 0:00:00 a 23:59:59
Decimal	128	1.0E-28 a 7.9E+28
Short	16	-32,768 a 32,767
Integer	32	-2,147,483,648 a 2,147,483,647
Long	64	-9,223,372,036,854,775,808 a 9,223,372,036,854,775,807
Single	32	1.5E-45 a 3.4E+38
Double	64	5.0E-324 a 1.7E+308
Object	32	Dado de qualquer tipo
String		0 a ~2000000000 caracteres Unicode

# Duração de Identificadores

- Duração de identificador
  - Período no qual um identificador existe na memória
- Escopo do identificador
  - Porção do programa na qual o identificador da variável pode ser referenciado
- Duração automática
  - Identificador que representa variáveis locais em um procedimento tem duração automática
- Variável de Instância
  - Uma variável declarada em uma classe
  - Elas existem enquanto a classe contendo tal variável estiver carregada na memória

# Regras de Escopo

- Escopos possíveis
  - Escopo de classe
    - Começa na identificação da classe após a palavra-chave **Class** e termina no comando **End Class**
  - Escopo de Módulo
    - Variável declarada no módulo tem escopo de módulo, o qual é similar ao escopo de classe
  - Escopo de nome de espaço
    - Procedimentos definidos em um módulo tem escopo de nome de espaço, o que geralmente significa que eles tem acesso durante todo o projeto
  - Escopo de bloco
    - Identificadores declarados dentro de um bloco, como o corpo de um procedimento ou o corpo de uma seleção If/Then, tem escopo de bloco

# Exemplo de Escopo de Variáveis

```
Public Class FormCombustivel
    Dim mediageral As Double

    .
    .
    .
End Class
```

Variável mediageral  
tem escopo de classe

```
Private Sub BtnCalcular_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnCalcular.Click
```

```
    Dim km, lt As Double
```

```
    .
    .
    .
```

```
End Sub
```

Variáveis km e lt tem  
escopo de bloco

# Exercício

**Cálculo de Consumo de Combustível**

Dados de Entrada

Distância Percorrida:  Km

Quantidade Abastecida:  litros

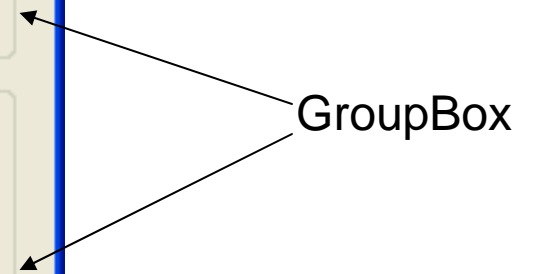
Consumo Calculado

Consumo Médio:  Km/litro

Média Geral:  Km/litro

Total Percorrido:  Km

Total Abastecido:  litros



# Exercício

- Implementar o programa de cálculo de consumo de combustível no ambiente gráfico
- Requisitos:
  - O programa deve verificar se o usuário digitou apenas números nas caixas de entrada e tratar o erro caso necessário
  - O programa deve evitar divisões por zero
  - Ao clicar no botão “Calcular” a média de consumo desse abastecimento é exibida na caixa “Consumo Médio”
  - “Média Geral”, “Total Percorrido” e “Total Abastecido” devem atualizar seus totais a cada clique em “Calcular”
  - “Média Geral”, “Total Percorrido” e “Total Abastecido” devem ser zerados quando o programa é iniciado e quando o usuário clica no botão “Limpar Totais”
  - Os números devem ser apresentados alinhados à direita
    - Dica: utilize a propriedade TextAlign
  - O usuário deve ser capaz de operar o programa usando apenas o teclado (verifique o funcionamento correto da tecla TAB e das teclas de atalho para os botões)
    - Dica: utilize a propriedade TabIndex

# Exercício 2

**Cálculo de Consumo de Combustível**

Dados de Entrada

Distância Percorrida:  Km

Quantidade Abastecida:  litros

Gasolina  Álcool

Consumo Calculado

Consumo Médio:  Km/litro

Média Geral:  Km/litro

Total Percorrido:  Km

Total Abastecido:  litros

RadioButtons

# Exercício 2

- Modifique o programa anterior para:
  - Permitir que o usuário selecione qual o tipo de combustível é utilizado:
    - Gasolina
    - Álcool
  - Mostre a caixa de “Consumo Médio” com fundo em vermelho quando ela estiver com valor:
    - Menor que 12 Km/litro para carros a gasolina
    - Menor que 8 Km/litro para carros a álcool
  - E em verde caso contrário
  - Repita o procedimento para a caixa “Média Geral”

# Referências Bibliográficas

- MSDN: <http://msdn.microsoft.com/vstudio/>
- DEITEL, Harvey M.; DEITEL, Paul J.; NIETO, Tem R. *Véual Basic.NET: Como Programar*. Prentice-Hall, 2004