

Escola Técnica Estadual “São Paulo” - ETESP
Linguagem C# - Visual Studio 2019

Estrutura condicional: Switch/Case

Exe016 → A partir da seleção de uma cidade brasileira, o aplicativo deverá exibir o “DDD” correspondente.

Vamos a uma solução “preliminar” (e resumida com alguns poucos DDDs) utilizando a estrutura condicional “if...else”:

DDD	CIDADE/ESTADO
11	São Paulo /SP
19	São Pedro/SP
19	Americana/SP
19	Campinas/SP
21	Rio de Janeiro / RJ
27	Vitória / ES
31	Belo Horizonte / MG
41	Curitiba / PR
48	Florianópolis / SC
51	Porto Alegre / RS
61	Brasília / DF
62	Goiania / GO
63	Palmas / TO

OBS: Estão destacadas, em “Vermelho”, três cidades com o mesmo “DDD”

INTERFACE GRÁFICA

The screenshot shows a Windows application titled "DDD - Cidades Brasileiras". The interface has a light gray background. At the top, there's a title bar with standard Windows window controls. Below the title bar, the text "DDD Cidades Brasileiras" is centered. The main area is divided into two sections. The top section, titled "Selecione a Cidade", has a light blue background. It contains a dropdown menu showing "Americana/SP" and a button labeled "Pesquisar DDD". The bottom section, titled "Resultado da Pesquisa:", has a light yellow background. It displays "DDD --> 19" where "19" is highlighted in a light orange box. To the right of this section are two buttons: "Limpar" and "SAIR". On the left side of the application, there are four annotations in white boxes with black text, each with an arrow pointing to a specific UI element: "GroupBox" points to the "Selecione a Cidade" section; "ComboBox 'CboCidade'" points to the dropdown menu; "GroupBox" points to the "Resultado da Pesquisa:" section; and "Label 'LbIDDD'" points to the "19" in the result display.

Prof. Roberto de Castro

Utilizando a estrutura condicional “if...else”, a programação do botão “Pesquisar”, poderá ser algo do tipo:

```
private void BtnPesquisar_Click(object sender, EventArgs e)
{
    if (CboCapital.Text == "São Paulo/SP")
    {
        LblDDD.Text = "11";
    }
    else if (CboCapital.Text == "Rio de Janeiro/RJ")
    {
        LblDDD.Text = "21";
    }
    else if (CboCapital.Text == "Vitória/ES")
    {
        LblDDD.Text = "27";
    }
    //else if
    //
    else if (CboCapital.Text == "Palmas/TO")
    {
        LblDDD.Text = "63";
    }
}
```

Percebe-se que ao longo do bloco “if” temos uma da condição que se repete. Estamos SEMPRE comparando a mesma informação “CboCapital.Text”, com outros valores!!!

Para estes casos, a estrutura condicional “switch....case” é a mais indicada para a solução do problema.

Sintaxe:

```
switch(VARIAVEL)
{
    case VALOR:
        //Faz algo se VARIAVEL for igual ao VALOR
        break;
    default: //(OPCIONAL)
        //Faz algo se VARIAVEL não for igual a nenhum CASE
        break;
}
```

IMPORTANTE: A instrução “switch” só pode ser utilizada em tipos como “int” ou “string”. Com qualquer outro tipo (float, double...), deveremos utilizar a estrutura condicional “if”!!!!

SOLUÇÃO UTILIZANDO A ESTRUTURA SWITCH....CASE

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

```
namespace Exe016
```

2- Revisão 7/7/2022

```
{
public partial class FrmExe016 : Form
{
    public FrmExe016()
    {
        InitializeComponent();
    }

    private void FrmExe016_Load(object sender, EventArgs e)
    {
        //OBS: Propriedade SORTED está TRUE
        // Propriedade DropDownStyle = DropDownList

        CboCidade.Items.Add("São Paulo/SP");
        CboCidade.Items.Add("São Pedro/SP");
        CboCidade.Items.Add("Americana/SP");
        CboCidade.Items.Add("Campinas/SP");
        CboCidade.Items.Add("Rio de Janeiro/RJ");
        CboCidade.Items.Add("Vitória/ES");
        CboCidade.Items.Add("Belo Horizonte/MG");
        CboCidade.Items.Add("Curitiba/PR");
        CboCidade.Items.Add("Florianópolis/SC");
        CboCidade.Items.Add("Porto Alegre/RS");
        CboCidade.Items.Add("Brasília/DF");
        CboCidade.Items.Add("Goiania/GO");
        CboCidade.Items.Add("Palmas/TO");

        CboCidade.SelectedIndex = 0;
    }

    private void BtnPesquisar_Click(object sender, EventArgs e)
    {
        string cidade = CboCidade.Text;

        switch (cidade)
        {
            case "São Paulo/SP":
                LblDDD.Text = "11";
                break;
            case "São Pedro/SP":
            case "Americana/SP":
            case "Campinas/SP":
                LblDDD.Text = "19";
                break;
            case "Rio de Janeiro/RJ":
                LblDDD.Text = "21";
                break;
            case "Vitória/ES":
                LblDDD.Text = "27";
                break;
        }
    }
}
```

```
case "Belo Horizonte/MG":  
    LblDDD.Text = "31";  
    break;  
case "Curitiba/PR":  
    LblDDD.Text = "41";  
    break;  
case "Florianópolis/SC":  
    LblDDD.Text = "48";  
    break;  
case "Porto Alegre/RS":  
    LblDDD.Text = "51";  
    break;  
case "Brasília/DF":  
    LblDDD.Text = "61";  
    break;  
case "Goiânia/GO":  
    LblDDD.Text = "62";  
    break;  
default:  
    LblDDD.Text = "63";  
    break;  
}  
}
```

```
private void BtnSair_Click(object sender, EventArgs e)  
{  
    Application.Exit();  
}
```

```
private void BtnLimpar_Click(object sender, EventArgs e)  
{  
    LblDDD.Text = "";  
    CboCidade.SelectedIndex = 0;  
    BtnPesquisar.Focus();  
}  
}  
}
```

Prof. Roberto de Castro

Estruturas de repetição

Também conhecidas como estruturas de looping, permitem a execução de um bloco de instruções repetidamente até que uma condição seja verdadeira.

1 - While → Sintaxe da estrutura de repetição “While”:

```
while (condição)
{
    comandos;
}
```

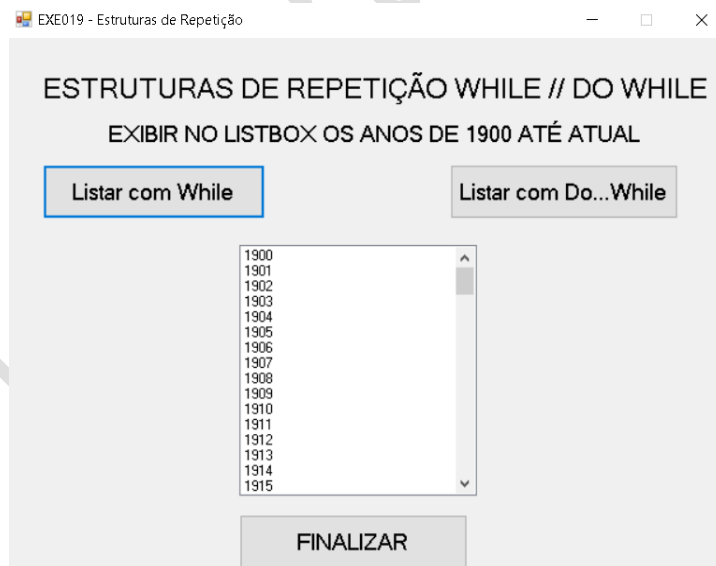
IMPORTANTE: O teste “condicional” é efetuado ANTES da execução do(s) comando(s).

2 - “Do While”: Sintaxe da estrutura de repetição “do While”:

```
do
{
    comandos;
}
while (condição);
```

IMPORTANTE: Ao contrário da estrutura “while”, o teste de “condição” é efetuado **após** a execução dos comandos. O que significa que o bloco de comandos será executado pelo menos uma vez.

Veremos no exemplo abaixo, o mesmo tipo de problema, porém, com duas soluções diferentes:



A solução apresenta os recursos da estrutura de repetição, afinal, seria uma tarefa bem árdua fazer esta programação exibindo os anos “um por um”...

Veja a programação:

SOLUÇÃO

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

Prof. Roberto de Castro

namespace Exe019

```
{
    public partial class Frm019 : Form
    {
        public Frm019()
        {
            InitializeComponent();
        }

        private void BtnSair_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void BtnListarWhile_Click(object sender, EventArgs e)
        {
            LstAnos.Items.Clear();
            int anoAtual = 0;
            anoAtual = Convert.ToInt16(DateTime.Now.ToString("yyyy"));
            int anoInicial = 1900;
            while (anoInicial <= anoAtual)
            {
                LstAnos.Items.Add(anoInicial);
                anoInicial++; //incrementa 1 no anoInicial
            }
        }

        private void BtnListarDoWhile_Click(object sender, EventArgs e)
        {
            LstAnos.Items.Clear();
            int anoAtual = 0;
            anoAtual = Convert.ToInt16(DateTime.Now.ToString("yyyy"));
            int anoInicial = 1900;
            do
            {
                LstAnos.Items.Add(anoInicial);
                anoInicial++; //incrementa 1 no anoInicial
            }
            while (anoInicial <= anoAtual);
        }
    }
}
```

Os valores exibidos no "ListBox", com as duas propostas, serão os mesmos, mas, para poder perceber a diferença entre uma estrutura e outra, vamos "forçar" e "fixar" o valor inicial da variável "anoAtual" com o número 1800. Faça a alteração e execute o problema.

Perceberemos que ao clicar no botão "Exibir com While" nenhum valor será exibido no ListBox, porém, quando clicamos no botão "Exibir com Do...While", o aplicativo exibirá 1900. Por que isto ocorre?

Dúvida: O que ocorrerá e a instrução "anoInicial++" não for utilizada?? Faça o teste.

3 - Estrutura de repetição “For”: A sintaxe da estrutura “For” possui três parâmetros que devem ser fornecidos para que funcione corretamente. O bloco de comandos será executado até que a condição proposta seja satisfeita. Sintaxe da estrutura:

```
For (<início>;<condição>;incremento>)  
{  
    comandos;  
}
```

Vamos utilizar o mesmo projeto anterior, agora com mais esta solução, vamos incluir um novo botão: “Listar com For”. Veja a programação:

```
private void BtnListarFor_Click(object sender, EventArgs e)  
{  
    LstAnos.Items.Clear();  
  
    int anoAtual = 0;  
    anoAtual = Convert.ToInt16(DateTime.Now.ToString("yyyy"));  
  
    for (int anoInicial = 1900; anoInicial <= anoAtual; anoInicial++)  
    {  
        LstAnos.Items.Add(anoInicial);  
    }  
}
```

A estrutura “For” apresenta uma solução mais enxuta, pois incorpora no próprio cabeçalho:

- a declaração da variável e o valor inicial desta variável (int anoInicial=1900);
- a condição em que o “loop” deverá ser executado (anoInicial <= anoAtual) e
- o incremento na variável “anoInicial”.

Mas precisamos ficar atentos pois esta estrutura não servirá para o caso em que a condição final não for uma expressão numérica.... Veremos mais exemplos!!

Prof. Roberto de Castro

Exercício proposto Exe020: Tabuada

INTERFACE

TABUADA

Selecione o Número

3

CALCULAR LIMPAR SAIR

Resultados

3 * 0 = 0
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30

SOLUÇÃO

```
using System;

using System.Windows.Forms;

namespace Exe020
{
    public partial class FrmExe020 : Form
    {
        public FrmExe020()
        {
            InitializeComponent();
        }

        private void FrmExe020_Load(object sender, EventArgs e)
        {
            //Preenche o Combo
            for (int x = 1; x <= 10; x++)
            {
                CboNumeros.Items.Add(x);
            }

            CboNumeros.SelectedIndex = 0;
        }
    }
}
```


Prof. Roberto de Castro

```
private void BtnCalcular_Click(object sender, EventArgs e)
{
    LstResultados.Items.Clear();
    int resultado = 0;

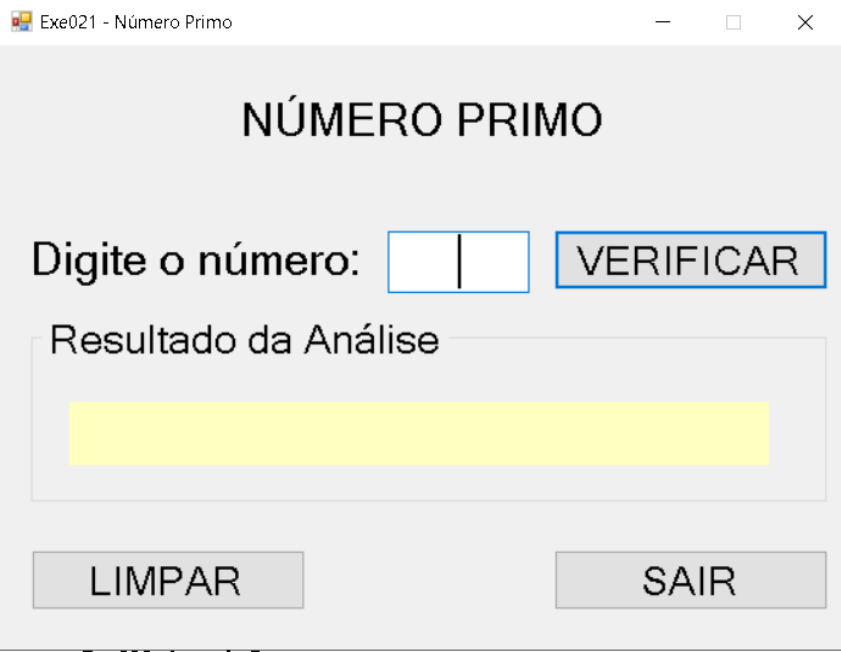
    for (int contador = 0; contador <=10; contador++)
    {
        resultado = contador * Convert.ToInt16(CboNumeros.SelectedItem);
        LstResultados.Items.Add(CboNumeros.SelectedItem + " * " + contador + " = " +
            resultado);
    }
}

private void BtnSair_Click(object sender, EventArgs e)
{
    Application.Exit();
}

private void BtnLimpar_Click(object sender, EventArgs e)
{
    LstResultados.Items.Clear();
    CboNumeros.SelectedIndex = 0;
}
}
```

Exercício proposto Exe021: Número primo

INTERFACE



Definição: Número “primo” é o número natural (inteiro e positivo) que pode ser dividido por apenas dois fatores: pelo **número** um e por ele mesmo. É importante perceber que o **número** um não é considerado um **número primo**, porque ele é divisível apenas por ele mesmo. Por outro lado, o **número** dois é o único **número primo** que também é um **número** par. Portanto, todos os demais números **primos** dessa sequência numérica considerada infinita serão obrigatoriamente ímpares. E o **número** 0, é **primo**? Utilizando a

Prof. Roberto de Castro

mesma definição, a resposta continua a ser não. Já que um **número primo** é divisível por ele próprio e **zero** não pode ser dividido por **zero**, já que é uma indeterminação. (fonte: <https://www.coc.com.br>).

Alguns números primos para teste: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, ...

SOLUÇÃO

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

namespace Exe021
{
    public partial class FrmExe021 : Form
    {
        public FrmExe021()
        {
            InitializeComponent();

            private void BtnVerificar_Click(object sender, EventArgs e)
            {
                LblResultado.Text = "";
                int numero = Convert.ToInt16(TxtNumero.Text);
                int resto = 0;

                if (numero <= 1)
                {
                    LblResultado.Text = "Número não é primo!!";
                    return;
                }

                for (int divisor = 2; divisor < numero; divisor++)
                {
                    //Devolve o resto da divisão
                    resto = numero % divisor;

                    if (resto == 0)
                    {
                        LblResultado.Text = numero.ToString() + " não é primo!!";
                        //ou
                        //LblResultado.Text = Convert.ToString(numero) + " não é primo!!";

                        //finaliza o loop --> break
                        //ou finaliza o procedimento
                        return;
                    }
                }

                LblResultado.Text = numero.ToString() + " é um número primo!!";
            }
        }
    }
}
```

```
private void BtnSair_Click(object sender, EventArgs e)
{
    Application.Exit();
}

private void TxtNumero_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsDigit(e.KeyChar) && e.KeyChar != 8)
    {
        e.Handled = true;
    }
}

private void BtnLimpar_Click(object sender, EventArgs e)
{
    LblResultado.Text = "";
    TxtNumero.Text = "";
    TxtNumero.Focus();
}
}
```