

Escola Técnica Estadual “São Paulo” - ETESP

Linguagem C# - Visual Studio 2019

Array (Vetores / Matrizes)

Em programação um **arranjo** (em inglês **array**) é uma estrutura de dados que armazena uma coleção de elementos de tal forma que cada um dos elementos possa ser identificado por, pelo menos, um índice ou uma chave. Essa estrutura de dados também é conhecida como **variável indexada**, **vetor** (para arranjos unidimensionais) e **matriz** (para arranjos bidimensionais). Os arranjos mantêm uma série de elementos de dados, geralmente do mesmo tamanho e tipo de dados.

(fonte: [https://pt.wikipedia.org/wiki/Arranjo_\(computação\)](https://pt.wikipedia.org/wiki/Arranjo_(computação)))

Em resumo: recurso que proporciona criar uma coleção de dados do mesmo tipo. A sintaxe para se declarar um array:

```
tipo[] nome = new tipo[]; // Declara um Array com uma dimensão.
```

```
tipo[,] matriz = new tipo[,]; // Para declarar um Array bidimensional (Matriz).
```

Exemplos:

- 1) Criação de um array com uma dimensão e com cinco (5) elementos numéricos inteiros:

```
int[] listaNumeros = new int[5];
```

Para “atribuir” os conteúdos para cada elemento do array:

```
listaNumeros[0] = 10;  
listaNumeros[1] = 15;  
listaNumeros[2] = 20;  
listaNumeros[3] = 25;  
listaNumeros[4] = 30;
```

IMPORTANTE: O número que está entre os colchetes chamamos de “índice”. É através dele que faremos referência a um elemento dentro do array.

Observe que o primeiro elemento do array começa no índice “zero”!!!

- 2) Criação de um array com cinco (5) elementos numéricos inteiros, com atribuição do conteúdo na mesma linha:

```
int[] listaNumeros={10, 15, 20, 25, 30};
```

- 3) Criação de um array com o número de elementos “variável” (representado por uma variável):

```
int x = 5;
```

```
int[] listaNumeros = new int[x];
```

- 4) Criação de um array com duas dimensões (matriz): 5 linhas e duas colunas, para implementar o modelo de tabela abaixo:

BD	DS
B	MB
R	B
I	R
MB	MB
B	B

```
string[,] resultados = new string[5,2];
```

Prof. Roberto de Castro

Para “atribuir” os conteúdos para cada elemento do array - matriz:

```
resultados[0,0] = "B";  
resultados[1,0] = "R";  
resultados[2,0] = "I";  
resultados[3,0] = "MB";  
resultados[4,0] = "B";
```

```
resultados[0,1] = "MB";  
resultados[1,1] = "B";  
resultados[2,1] = "R";  
resultados[3,1] = "MB";  
resultados[4,1] = "B";
```

Exemplos de utilização do Array: Exibir os cinco elementos do array em um “listBox”.

//Declaração de um array com 5 elementos numéricos inteiros

```
int[] listaNumeros = new int[5];
```

//Atribuição de conteúdo para cada elemento do array

```
listaNumeros[0] = 10;  
listaNumeros[1] = 15;  
listaNumeros[2] = 20;  
listaNumeros[3] = 25;  
listaNumeros[4] = 30;
```

//Exemplo 1: Rotina para exibir o conteúdo de cada elemento do Array em um “listBox”:

```
for (int x = 0; x <= 4; x++)  
{  
    listBox1.Items.Add(listaNumeros[x]);  
}
```

//Exemplo 2: Rotina para exibir o conteúdo de cada elemento do Array

// Importante: listaNumeros.Length retorna o número 5, pois o array possui 5 elementos, porém, a
// repetição SOMENTE poderá ocorrer enquanto x < 5!!!

```
for (int x = 0; x < listaNumeros.Length; x++)  
{  
    listBox1.Items.Add(listaNumeros[x]);  
}
```

**//Exemplo 3: Rotina para exibir o conteúdo de cada elemento do Array, utilizando a estrutura de
//repetição “foreach”, onde:**

//int numero representa UM elemento do array listaNumeros!!!

```
foreach (int numero in listaNumeros)  
{  
    listBox1.Items.Add(numero);  
}
```

Prof. Roberto de Castro

Exemplo de programa com Matriz : Exibir os elementos do array em um “listBox”.

BD	DS
B	MB
R	B
I	R
MB	MB
B	B

//Declaração de uma Matriz com 5 linhas e duas colunas

string [,] listaResultados = new string [5,2];

//Atribuição de conteúdo para cada elemento do array

//Menções de BD (Banco de Dados)

```
listaResultados[0,0] = "B";  
listaResultados[1,0] = "R";  
listaResultados[2,0] = "I";  
listaResultados[3,0] = "MB";  
listaResultados[4,0] = "B";
```

//Menções de DS (Desenvolvimento de Sistemas)

```
listaResultados[0,1] = " MB";  
listaResultados[1,1] = "B";  
listaResultados[2,1] = "R";  
listaResultados[3,1] = "MB";  
listaResultados[4,1] = "B";
```

//Exemplo 1: Rotina para exibir o conteúdo de cada elemento do Array

//Dúvida: Qual será o “tamanho deste array??” ao executar a instrução abaixo:

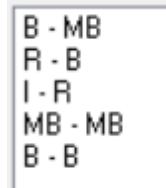
//MessageBox.Show(listaResultados.Length.ToString());

//Resposta → 10 !!!

//Preenchendo o listBox com as menções (por linha)

```
for (int linha = 0; linha <= 4; linha++)  
{  
    listBox1.Items.Add(listaResultados[linha, 0] + " - " + listaResultados[linha, 1]);  
}
```

Veja o resultado do listBox:



```
B - MB  
R - B  
I - R  
MB - MB  
B - B
```

Prof. Roberto de Castro

//Exemplo 2: Rotina para exibir o conteúdo de cada elemento do Array, utilizando o ForEa

```
foreach (string mencao in listaResultados)
{
    listBox1.Items.Add(mencao);
}
```

Veja o resultado do listBox:

B
MB
R
B
I
R
MB
MB
B
B

Interpretação do resultado com o ForEach: A matriz foi “percorrida” horizontalmente.

Veja o que diz a Microsoft:

“Em matrizes multidimensionais, os elementos são percorridos de modo a que os índices da dimensão mais à direita sejam aumentados primeiro e, em seguida, da próxima dimensão à esquerda, e assim por diante seguindo para a esquerda.

Com matrizes multidimensionais, usar um loop aninhado [for](#) oferece mais controle sobre a ordem na qual processar os elementos da matriz.”

Fonte: <https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/arrays/using-foreach-with-arrays>

Exercício “exemplo” Exe023: Receber um número entre 0 (zero) e 19 e exibir o “extenso do número”. Para este primeiro exercício, vamos resolver utilizando três técnicas distintas: If....else // Switch...Case e por último “array”.

INTERFACE GRÁFICA

Considerações:

- O ComboBox (CboNumero) será carregado com os números de 0 a 19, no Form_Load (propriedade DropDownStyle = DropDownList).
- O “Array” será do tipo “string”, pois armazenará os “extensos” dos números de 0 a 19.
- O “tamanho” do array será 20.
- Resumidamente, a utilização de “Array” prevê as etapas: A declaração do array e a atribuição do conteúdo dos elementos que compõem o array.

Prof. Roberto de Castro

- Com objetivo meramente didático, iremos declarar o array e atribuir o conteúdo diretamente no botão “Extenso com Array”. Importante destacar que esta solução criará o array e atribuirá conteúdo TODAS AS VEZES em que o botão for clicado. Este exercício será revisto em breve...

- Importante destacar que a solução com o “Array” utiliza o próprio número selecionado no “ComboBox” como índice do Array, isto somente é possível para estes casos em que a informação de origem é numérica e coincide com a posição do elemento dentro do array. Veremos em outros casos que esta técnica não será viável.

SOLUÇÃO

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

namespace Exe023
{
    public partial class FrmExe023 : Form
    {
        public FrmExe023()
        {
            InitializeComponent();
        }

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

        private void FrmExe023_Load(object sender, EventArgs e)
        {
            for (int x = 0; x <= 19; x++)
            {
                CboNumero.Items.Add(x);
            }
            CboNumero.SelectedIndex = 0;
        }

        private void BtnExibirIfElse_Click(object sender, EventArgs e)
        {
            int numero = Convert.ToInt16(CboNumero.SelectedItem);
            if (numero == 0)
            {
                LblExtenso.Text = "Zero";
            }
            else if (numero == 1)
            {
                LblExtenso.Text = "Um";
            }
            else if (numero == 2)
            {
                LblExtenso.Text = "Dois";
            }
            //...
            //...
            else if (numero == 18)
            {
                LblExtenso.Text = "Dezoito";
            }
            else
            {
            }
        }
    }
}
```

```
        LblExtenso.Text = "Dezenove";
    }
}

private void BtnExibirSwitch_Click(object sender, EventArgs e)
{
    int numero = Convert.ToInt16(CboNumero.SelectedItem);

    switch (numero)
    {
        case 1:
            LblExtenso.Text = "Um";
            break;

        case 2:
            LblExtenso.Text = "Dois";
            break;

        case 3:
            LblExtenso.Text = "Três";
            break;
        //...
        //...
        case 18:
            LblExtenso.Text = "Dezoito";
            break;

        default:
            LblExtenso.Text = "Dezenove";
            break;
    }
}

private void BtnExibirArray_Click(object sender, EventArgs e)
{
    //Declaração do Array. São 20 elementos de 0 a 19
    string[] extenso = new string[20];

    //Atribuição do conteúdo do Array
    extenso[0] = "Zero";
    extenso[1] = "Um";
    extenso[2] = "Dois";
    extenso[3] = "Três";
    extenso[4] = "Quatro";
    extenso[5] = "Cinco";
    extenso[6] = "Seis";
    extenso[7] = "Sete";
    extenso[8] = "Oito";
    extenso[9] = "Nove";
    extenso[10] = "Dez";
    extenso[11] = "Onze";
    extenso[12] = "Doze";
    extenso[13] = "Treze";
    extenso[14] = "Quatorze";
    extenso[15] = "Quinze";
    extenso[16] = "Dezesseis";
    extenso[17] = "Dezessete";
    extenso[18] = "Dezoito";
    extenso[19] = "Dezenove";

    int indice = Convert.ToInt16(CboNumero.SelectedItem);
    LblExtenso.Text = extenso[indice];
}
```

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

Exercício proposto Exe024: Terminal de consulta “Resultado Final”. Para este projeto vamos considerar uma turma de 40 alunos. A consulta será realizada pelo número de chamada. Haverá uma lista contendo o resultado final de cada aluno, seguindo o modelo abaixo:

Número de Chamada	Resultado Final
01	Aprovado
02	Aprovado
03	Retido
04	Desistente
05	Aprovado
06	Aprovado
07	Retido
08	Recuperação
09	Aprovado
10	Retido
..	..
..	..
..	..
40	Aprovado

INTERFACE GRÁFICA DO PROJETO

Exe024

COLÉGIO MODELO

TERMINAL DE CONSULTA - RESULTADOS FINAIS

Número de Chamada:

RESULTADO FINAL

Prof. Roberto de Castro

Considerações:

- A solução definirá o Array na área de declaração das variáveis de classe e atribuirá o conteúdo no evento "Form_Load" (desta forma o Array é criado uma única vez e o seu conteúdo também é declarado uma única vez).
- o "número do aluno" será utilizado como índice do array. **Atenção:** a relação inicia-se em 1 e o array inicia-se em zero!!!
- O tratamento da digitação do número através da estrutura "Try..Catch". O erro será gerado no caso da digitação de números fora da faixa 1 - 40 ou digitação de informação não numérica!!

Exercício proposto Exe025: A partir de uma lista de "40 Classificados no Vestibulinho" o projeto deverá: Receber o número de inscrição, verificar se consta na lista e informar se o candidato está classificado ou não.

LISTA DE CLASSIFICADOS	
Número de Ordem	(Número de Inscrição)
1	10514
2	30343
3	8240
4	3125
5	50525
6	23289
7	7310
8	9281
9	49524
10	33001
11	32
...	
...	
40	39720

Considerações:

- Se o número de inscrição existir na "lista de classificados", indicará que o candidato foi classificado;
- A "posição" dentro da lista, indica qual é a "classificação" do candidato no vestibulinho..
- Se o número de inscrição NÃO existir na "lista de classificados", exibir a mensagem "Lista de Espera".

Prof. Roberto de Castro

- Importante destacar que para a solução deste problema NÃO poderemos utilizar diretamente o “número de inscrição” como índice do array.

- Prováveis “soluções”: Utilizando-se a estrutura de repetição “for” (para reforçar o conteúdo previamente estudado), para fazer a pesquisa no array e outra solução utilizando-se recursos que o próprio “array” possui (Array.IndexOf).

INTERFACE GRÁFICA

The image shows a Windows application window titled "Exe025". The main content area has a light blue background and contains the following elements:

- Centered text: "ESCOLA TÉCNICA ESTADUAL ETESP" and "CONSULTA RESULTADO VESTIBULINHO".
- A label "NÚMERO DE INSCRIÇÃO:" followed by a text input field divided into two parts by a vertical line.
- Two buttons: "Consultar I" (light gray) and "Consultar II" (light blue with a blue border).
- A label "Status:" followed by a large, empty, light blue rectangular box.
- Two buttons at the bottom: "Limpar" (light gray) and "Sair" (light gray).

A large, diagonal watermark reading "Prof. Roberto de Castro" is visible across the lower half of the image.

Prof. Roberto de Castro

Informações complementares

Paralelamente ao conceito de “Array” podemos incluir as “Listas”, que para este caso, torna-se obrigatório a inclusão do namespace **“using System.Collections.Generic;”**.

Veja o exemplo abaixo:

```
//Declaração da lista
List<string> listaNomes = new List<string>();

//Conteúdo da lista
listaNomes.Add("Nome 0");
listaNomes.Add("Nome 2");
listaNomes.Add("Nome 3");

//exibe a lista
foreach (string nome in listaNomes)
{
    listBox1.Items.Add(nome);
}
```

O “Add” insere o elemento um após o outro.

Em um array NÃO conseguimos excluir/incluir um elemento, diferentemente de uma “List” que possui esta funcionalidade de inclusão/exclusão de elementos. Veja os exemplos:

Inserir → listaNomes.Insert(1, "Nome 1");

Irá inserir o "Nome 1" na posição 1, abaixo de Nome 0 (o índice de "Nome 0" é Zero!!!)

Excluir → listaNomes.Remove("Nome 3");

IMPORTANTE → o método REMOVE caso não encontre o elemento na lista, simplesmente será ignorado, por exemplo: **listaNomes.Remove("Nome 333");**

RemoveAt → para remover um elemento, dada a sua posição. Por exemplo: remover o elemento de posição Zero (Nome 0) → **listaNomes.Removeat(0);**

Para obter o número de elementos em uma “list” é o “count”. Por exemplo: **listaNomes.Count.**

Uma “List” possui uma DESVANTAGEM com relação ao Array que é o acesso direto ao elemento na posição “X”, sem precisar passar pelos anteriores. Na List, o acesso é SEQUENCIAL!!!

Faça um teste com o código abaixo:

```
//Declaração da lista
List<string> listaNomes = new List<string>();

//Conteúdo da lista
listaNomes.Add("Nome 0");
listaNomes.Add("Nome 2");
listaNomes.Add("Nome 3");

//exibe a lista
foreach (string nome in listaNomes)
{
    listBox1.Items.Add(nome);
}

MessageBox.Show("pausa");
listBox1.Items.Clear();
```

```
//Irá inserir o "Nome 1" na posição 1, abaixo de Nome 0 (o índice de "Nome 0"
    é Zero!!!)
listaNomes.Insert(1, "Nome 1");
foreach (string nome in listaNomes)
{
    listBox1.Items.Add(nome);
}

MessageBox.Show("pausa");
listBox1.Items.Clear();

//Irá remover o "Nome 3"
listaNomes.Remove("Nome 3");
foreach (string nome in listaNomes)
{
    listBox1.Items.Add(nome);
}

MessageBox.Show(listaNomes.Count.ToString());

MessageBox.Show("pausa");
listBox1.Items.Clear();

//Irá remover o elemento da posicao 2
listaNomes.RemoveAt(2);
foreach (string nome in listaNomes)
{
    listBox1.Items.Add(nome);
}
```