

# Aulas 11,12

- Vectors e Matrizes
- Métodos de Manipulação da Classe String

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3




1,1 →

2,3 →

3,2 →



# Introdução aos Arrays Unidimensionais

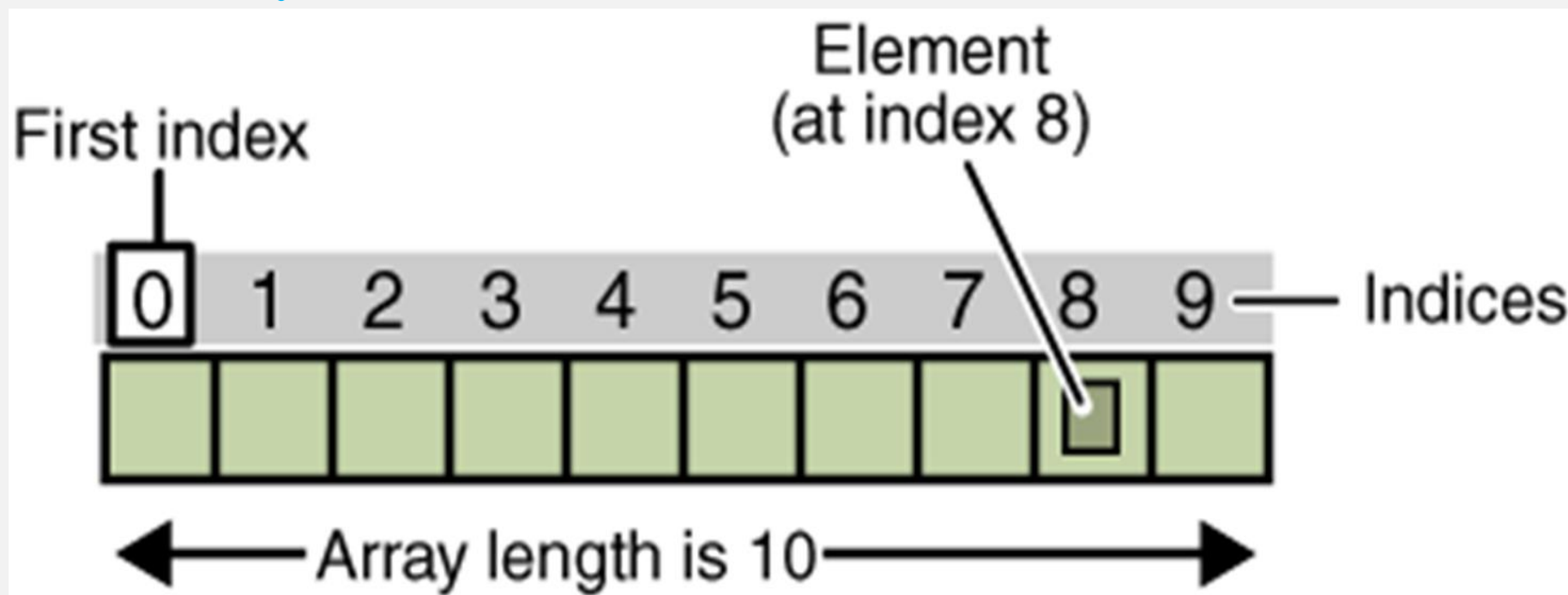
Uma **variável simples**, é uma variável que apenas pode conter um único valor.

E uma **variável composta** é aquela que pode conter dois ou mais valores.

Em **C#** as variáveis compostas são chamadas de **arrays**. Vamos estudar 2 tipos de arrays que são: arrays Unidimensionais, também chamados de **vetores** e arrays Bidimensionais, também chamados de **matrizes**.

# Vetores em C#

Um **vector** é uma variável composta homogênea unidimensional que dentro dela podemos armazenar valores do mesmo tipo.





# Declaração de Vetores em C#

Como C# é orientado á Objectos, os Vetores em C# são objectos da classe do tipo de dados que pretendemos (Classe Array)

Para declarar um vetor usamos a seguinte sintaxe:

```
tipo_de_dado[] nome_do_vetor = new tipo_de_dado[tamanho];
```

Ex: declare um vetor que contenha 10 nomes.

```
string[] nomes = new string[10];
```

# Inicialização de Vetores em C#

Em C# é possível declarar um vetor e ao mesmo tempo inicializá-lo.

**Ex:** Declare um vetor com as 5 vogais do alfabeto:

**1ª resolução:** `char[] vogais = new char[5]{'a', 'e', 'i', 'o', 'u'};`

**2ª resolução:** `char[] vogais = new char[5];`

`vogais[0]='a';`

`vogais[1]='e';`

`vogais[2]='i';`

`vogais[3]='o';`

`vogais[4]='u';`

# Propriedades de Vetores em C#

Como os Vetores são Objectos da Classe Array, eles herdam alguns atributos desta classe. São eles:

<b>IsFixedSize</b>	Retorna um valor indicando se um array possui um tamanho fixo ou não.
<b>IsReadOnly</b>	Retorna um a valor indicando se um array é somente-leitura ou não.
<b>IsSynchronized</b>	Retorna um a valor que indica se o acesso a um array é thread-safe ou não.
<b>Length</b>	Retorna o número total de itens em todas as dimensões de um array
<b>Rank</b>	Retorna o número de dimensões de um array
<b>SyncRoot</b>	Retorna um objeto que pode ser usado para sincronizar o acesso a um array.



# Métodos de Manipulação de Vetores em C#

<b>BinarySearch</b>	Procura em um array unidimensional ordenado por um valor usando o algoritmo de busca binário.
<b>Clear</b>	Remove todos os itens de um array e define um intervalo de itens no array com valor zero.
<b>Clone</b>	Cria uma cópia do Array.
<b>Copy</b>	Copia uma seção de um array para outro array e realiza a conversão de tipos e boxing requeridas.
<b>CopyTo</b>	Copia todos os elementos do array unidimensional atual para o array unidimensional especificado iniciando no índice de destino especificado do array.
<b>CreateInstance</b>	Inicializa uma nova instância da classe Array.
<b>GetEnumerator</b>	Retorna um IEnumerator para o Array.
<b>GetLength</b>	Retorna o número de itens de um Array.

# Métodos de Manipulação de Vetores em C#

<b>GetLowerBound</b>	Retorna o primeiro item de um Array.
<b>GetUpperBound</b>	Retorna o último item de um Array.
<b>GetValue</b>	Retorna o valor do item especificado no Array.
<b>IndexOf</b>	Retorna o índice da primeira ocorrência de um valor em um array de uma dimensão ou em uma porção do Array.
<b>LastIndexOf</b>	Retorna o índice da última ocorrência de um valor em um array unidimensional ou em uma porção do Array.
<b>Reverse</b>	Reverte a ordem de um item em um array de uma dimensão ou parte do array.
<b>SetValue</b>	Define o item especificado em um array atual para o valor definido.
<b>Sort</b>	Ordena os itens de um array.



# Exemplos de uso de Propriedades e Métodos

```
string[] Cores = { "vermelho", "verde", "amarelo", "laranja", "azul" };
```

<b>Obter o tamanho do Array</b>	<code>Cores.Length;</code>
<b>Ordenar o Array</b>	<code>Array.Sort(Cores)</code>
<b>Inverter a ordem dos itens no Array</b>	<code>Array.Reverse(Cores)</code>
<b>Usar GetLowerBound/GetUpperBound</b>	<pre>for( int j = Cores.GetLowerBound(0); j &lt;= Cores.GetUpperBound(0); j++) {     listBox2.Items.Add("Cores[0] = " + j + " " + Cores[j]); }</pre>

# Exemplos de uso de Propriedades e Métodos

<b>Verificando se o Array tem tamanho fixo</b>	<pre>if (Cores.IsFixedSize) {     listBox2.Items.Add("O array e fixo");     listBox2.Items.Add(" tamanho =&gt; (Cores.Lenght) = " + Cores.Length);     listBox2.Items.Add(" intervalo =&gt; (Cores.Rank) = " + Cores.Rank); }</pre>
<b>Realizando uma busca binária no Array</b>	<pre>object oCor = "verde";  int retorno = Array.BinarySearch(Cores, oCor); if(retorno &gt;=0)     listBox2.Items.Add("Indice do Item " + retorno.ToString()); else     listBox2.Items.Add("Item não localizado");</pre>
<b>Obtendo o índice de um item do Array</b>	<pre>int ind = Array.IndexOf(Cores, "verde"); listBox2.Items.Add("O índice do item 'verde' e " + ind);</pre>



# Exercícios Usando Vetores

Faça um programa que gere um Vector aleatoriamente. A quantidade do Vector é informada pelo usuário. Bem como o Limite mínimo e máximo dos Valores a serem gerados. Após isto, crie código que permita:

- a) Calcular o Maior número gerado
- b) Calcular a média entre os números pares gerados
- c) O número mais gerado e uma lista dos que foram gerados apenas uma vez, e os que foram gerados o número de vezes que o usuário quiser



# Exercícios Usando Vetores

Faça um programa que leia um Vector de Nomes e Ordene o Vector. Os Elementos do Vector serão ordenados um á um, como se alguém estivesse a ordená-los manualmente. Crie Código usando Timer de modos que as trocas de lugares sejam visíveis ao usuário. Calcule quanto Tempo em Segundos demorou para ordenar o Vector.

# Exercícios Usando Vetores

Uma matriz é uma variável composta homogênea bidimensional que dentro dela podemos armazenar valores do mesmo tipo.

Matriz M	Coluna 1	Coluna 2	Coluna 3
Linha 1	M[0][ 0]	M[0][1]	M[0][2]
Linha 2	M[1][0]	M[1][1]	M[1][2]
Linha 3	M[2][0]	M[2][1]	M[2][2]

Nome da matriz

Índice da linha

Índice da coluna





# Matrizes em C#

Para declarar uma matriz usamos a seguinte sintaxe:

```
tipo_de_dado[,] nome_da_matriz = new tipo_de_dado[nº de linhas,  
nº de colunas];
```

Ex: declare uma matriz de inteiros de ordem 3.

```
int[,] inteiros = new int[3,3];
```



# Declaração e Inicialização de Matrizes em C#

Em C# é possível declarar uma matriz e ao mesmo tempo inicializá-la.

**Ex:** Declare uma matriz 3X3 que contenha os números de 1 á 9

```
int[,] inteiros = new int[3,3] {{1,2,3}, {4,5,6}, {7,8,9}};
```

# Exemplos de Declaração e Inicialização de Matrizes em C#

<pre>int[,] numeros = new int[3, 2] { {1, 2}, {3, 4}, {5, 6} }; string[,] amigos = new string[2, 2] { {"Mac", "Jan"}, {"Mimi", "Jeff"} };</pre>	definindo o tamanho e o operador <b>new</b>
<pre>int[,] numeros = new int[, ] { {1, 2}, {3, 4}, {5, 6} }; string[,] amigos = new string[, ] { {"Mac", "Jan"}, {"Mimi", "Jeff"} };</pre>	omitindo o tamanho do array
<pre>int[,] numeros = { {1, 2}, {3, 4}, {5, 6} }; string[,] amigos = { {"Mac", "Jan"}, {"Mimi", "Jeff"} };</pre>	Omitindo o operador <b>new</b>



# O Laço Foreach para Vetores e Matrizes

O laço foreach pode ser usado para acessar cada elemento de um vector, de uma matriz ou de uma colecção.

A sintaxe usada é :

```
foreach (<tipo> <variavel> in <coleção/vector/Matriz>)
```

# Exemplo de Foreach para Vetores

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

```
numeros [0] = 10;
```

```
numeros [1] = 4;
```

```
numeros [2] = 32;
```

```
numeros [3] = 1;
```

```
numeros [4] = 20;
```

```
foreach (int i in numeros )
```

```
{
```

```
    listBox1.Items.Add("i = " + i);
```

```
}
```

Usando laço foreach

i = 10

i = 4

i = 32

i = 1

i = 20



# Exemplo de Foreach para Vetores de Strings

```
string[] semana = new string[7];
```

```
semana[0] = "Domingo";
```

```
semana[1] = "Segunda-feira";
```

```
semana[2] = "Terça-feira";
```

```
semana[3] = "Quarta-feira";
```

```
semana[4] = "Quinta-feira";
```

```
semana[5] = "Sexta-feira";
```

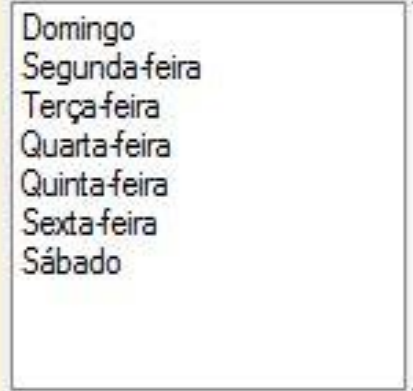
```
semana[6] = "Sabado";
```

```
foreach (string dia in semana)
```

```
{
```

```
    listBox1.items.Add(dia);
```

```
}
```

A rectangular box representing a ListBox, containing a list of the days of the week in Portuguese, one per line.

Domingo  
Segunda-feira  
Terça-feira  
Quarta-feira  
Quinta-feira  
Sexta-feira  
Sábado

# Exemplo de Foreach para Matrizes

```
int[,] numeros2 = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 }, { 9, 10 } };  
  
foreach (int i in numeros2 )  
{  
    listBox1.Items.Add("i = " + i);  
}
```

Usando laço foreach com vetor bidimensional

i= 1  
i= 2  
i= 3  
i= 4  
i= 5  
i= 6  
i= 7  
i= 8  
i= 9  
i= 10

**Tarefa:** Crie um Programa Semelhante, mas desta vez para imprimir a matriz em formato de linhas e colunas num objecto:

- a) ListBox
- b) Form



# Exercícios Usando Matrizes

Faça um programa que leia uma matriz 5x5 e imprima num datagridview.

A Seguir o Usuário vai escolher:

- a) A Linha que quer ver impressa numa listbox (os valores estão dispostos na horizontal).
- b) A Coluna que quer ver impressa (os Valores estão impressos na vertical)
- c) Pintar a Diagonal Principal com a Cor VERDE
- d) A soma de cada linha e guarde num vetor e imprima em outro datagridview.



# Exercícios Usando Matrizes

- e) Pintar a Diagonal Secundária com a Cor Azul
- f) A soma de todas as linhas e guardar num vector
- g) Soma de todas as colunas e Guardar num vector
- h) Ordenar a Matriz
- i) Apagar os elementos da matriz por linha
- j) Apagar os elementos da matriz por coluna



# Exercícios Usando Matrizes

Elabore um programa que fornece uma matriz  $C$  com o triângulo de pascal até a  $n$ -ésima linha (dado fornecido pelo usuário):

1

11

121

1331



# Exercícios Usando Matrizes

Ler uma Matriz  $L \times C$  e Imprimir a Sua Matriz Transposta

$$\text{a) } M = \begin{pmatrix} -1 & 8 \\ 2 & 5 \end{pmatrix} \Rightarrow M^t = \begin{pmatrix} -1 & 2 \\ 8 & 5 \end{pmatrix}$$

$$\text{b) } B = \begin{pmatrix} 1 & 7 & 12 \\ 4 & -4 & 6 \\ 9 & 0 & 5 \end{pmatrix} \Rightarrow B^t = \begin{pmatrix} 1 & 4 & 9 \\ 7 & -4 & 0 \\ 12 & 6 & 5 \end{pmatrix}$$

$$\text{c) } C = \begin{pmatrix} -3 & 6 & 2 \\ -1 & 0 & 7 \end{pmatrix} \Rightarrow C^t = \begin{pmatrix} -3 & -1 \\ 6 & 0 \\ 2 & 7 \end{pmatrix}$$



# Strings em C#

Uma String é um conjunto de caracteres. Em C# tudo que está entre aspas duplas é uma String.

Em C# strings são **IMUTÁVEIS**, ou seja uma vez declaradas e inicializadas não podem ser mudadas.

# Strings em C#

1- **ToLower()**: Converte todos os caracteres de uma string minúsculo.

Ex: `string nome = "LUCAS", nome2;`

`nome2 = nome.ToLower();`

2- **ToUpper( )**: Converte todos os caracteres de uma string maiúsculo.

3- **TrimStart( )**: Remove todos os espaços em branco no início da string.

4- **TrimEnd( )**: Remove todos os espaços em branco no fim da string.

5- **Trim( )**: Remove todos os espaços em branco no início e no fim da string.

# Strings em C#

6- **Substring(início, lenght):** retorna uma Substring começando de início contando lenght casas.

Ex: `string nome = "cultura", nome2;`

`nome2 = nome.Substring(3, 4); //nome2="tura"`

7- **Replace(old, new):** substitui na string o character old pelo character new.

Ex: `string nome = "ABONA", nome2;`

`nome2 = nome.Replace("b", "T"); //nome2="ATONA"`

# Strings em C#

- 8- **StartWith(character)**:Retorna true se a string inicia com o character e false se não inicia.
- 9- **EndWith(character)**:Retorna true se a string termina com o character e false se não termina.
- 10- **Contains(string)**:Retorna true se a string contém o argumento string.



# Strings em C#

- 11- **IndexOf(character)**: Retorna o índice onde o character está na string.
- 12- **LastIndexOf(character)**: Retorna a posição da última ocorrência do character na string.
- 13- **Split(character)**: Coloca os elementos de uma string e coloca em um vector separados por um character.



# Exercícios de Strings em C#

1-Faça um Programa que leia 2 nomes, de um homem e de uma mulher e imprima o novo nome da mulher, se eles contraíram matrimônio. Use a regra de que a mulher adota o sobrenome do Marido. Use sempre POO

2-Faça um Programa que leia um Nome Completo e abrevie os nomes do meio. Use sempre POO

Exemplo: Keyla Melanie Miguel Abel

Saída: Keyla M.M. Abel



# Exercícios de Strings em C#

## 3-Use a Mesma Classe para Resolver os seguintes problemas:

a) Faça um programa em C#.net que receba uma frase e permite criptografar esta frase. A criptografia consiste em substituir todas as vogais por #.

Exemplo:

Frase: Eu estou na Escola

Saída: ## #st## n# #sc#l#

b) Faça um programa em C#.net que se comporte como um vírus, ou seja repete cada palavra que o usuário inseriu na frase.

Exemplo

Frase: Eu estou na Escola

Saída: Eu Eu estou estou na na Escola Escola



# Exercícios de Strings em C#

c) Faça um programa que receba duas frases e gere uma terceira que represente a combinação das palavras das duas frases lidas.

Exemplo:

Frase 1: Hoje está um belo dia

Frase 2: Talvez chova amanhã

Saída : Hoje talvez está chova um amanhã belo dia

# Exercícios de Strings em C#

d) Faça um programa que receba duas frases e gere uma terceira que represente a combinação das palavras das duas frases lidas.

Exemplo:

Frase 1: Hoje está um belo dia

Frase 2: Talvez chova amanhã

Saída : Hoje talvez está chova um amanhã belo dia