

# Linguagem Programação

## II – Aula 03

Prof: Vinicius Drumond Gonzaga

# Operadores

- ▶ Aritméticos;
- ▶ Comparação;
- ▶ Lógicos;
- ▶ Operadores de Igualdade.

# Operadores Aritméticos (Binários)

Operador Aritmético	Descrição
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Módulo (resto da divisão)

# Operadores Aritméticos (Unários)

- ▶ Incremento ++
- ▶ Decremento --
- ▶ incremento pós-fixado
- ▶ incremento de prefixo
- ▶ decremento pós-fixado
- ▶ decremento de prefixo

# Exemplo incremento pós-fixado

- ▶ `int i = 3;`
- ▶ `Console.WriteLine(i);` // output: 3
- ▶ `Console.WriteLine(i++);` // output: 3
- ▶ `Console.WriteLine(i);` // output: 4

# Exemplo incremento de prefixo

- ▶ `double a = 1.5;`
- ▶ `Console.WriteLine(a);` // output: 1,5
- ▶ `Console.WriteLine(++a);` // output: 2,5
- ▶ `Console.WriteLine(a);` // output: 2,5

# Exemplo decremento pós-fixado

- ▶ `int i = 3;`
- ▶ `Console.WriteLine(i); // output: 3`
- ▶ `Console.WriteLine(i--); // output: 3`
- ▶ `Console.WriteLine(i); // output: 2`

# Exemplo decremento de prefixo

- ▶ `double a = 1.5;`
- ▶ `Console.WriteLine(a);` // output: 1.5
- ▶ `Console.WriteLine(--a);` // output: 0.5
- ▶ `Console.WriteLine(a);` // output: 0.5



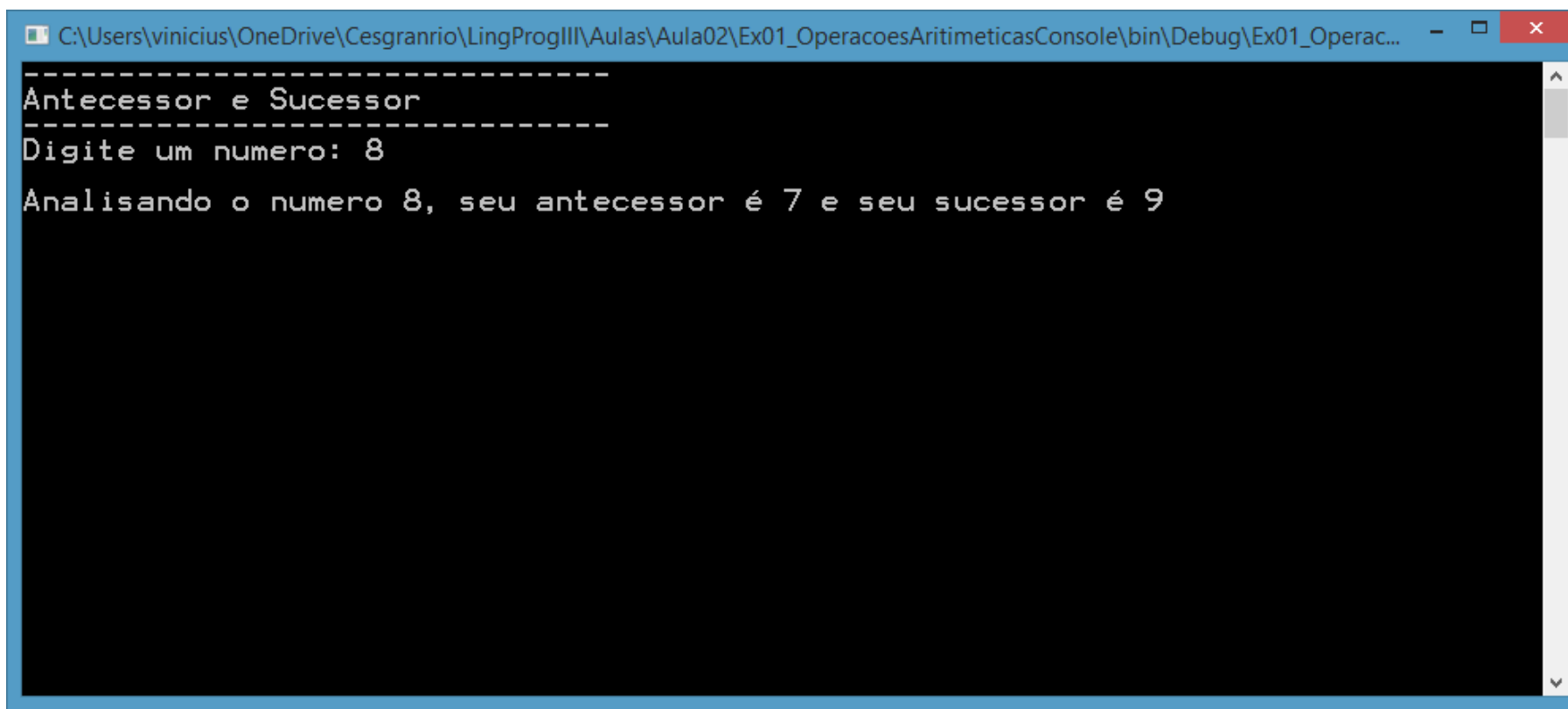
# Operadores Aritméticos (simplificando)

Operador Aritmético	Descrição
+ =	mais igual
- =	menos igual
* =	vezes igual
/ =	dividido igual
% =	módulo igual

# Exemplo

- ▶ `int x = 5;`
- ▶ `x += 5; // é a mesma coisa que x = x + 5`
- ▶ `Console.WriteLine("Valor do x = " + x);`
- ▶ `Console.ReadKey();`

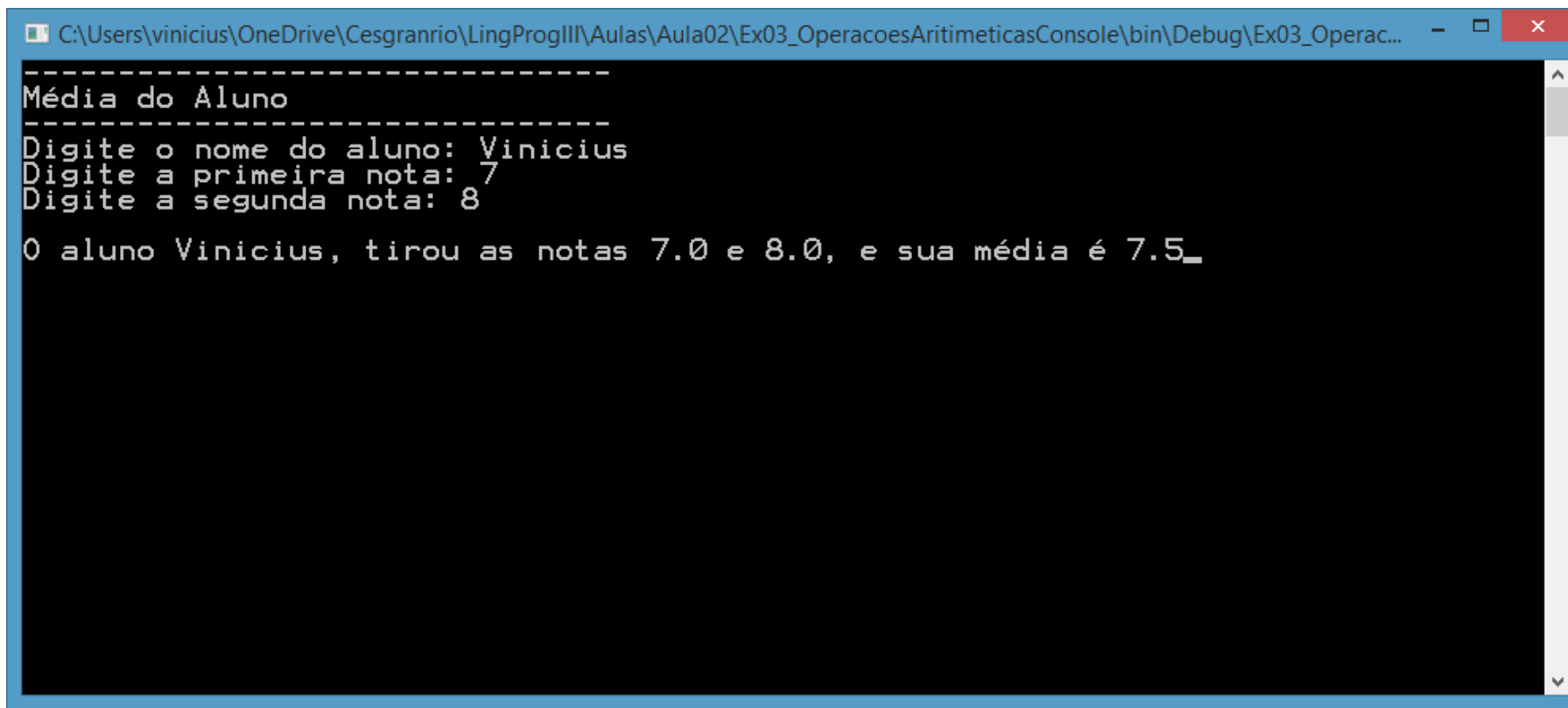
# Exercício 01



The screenshot shows a Windows console window with a blue title bar. The title bar text is "C:\Users\vinicius\OneDrive\Cesgranrio\LingProgIII\Aulas\Aula02\Ex01\_OperacoesAritimeticasConsole\bin\Debug\Ex01\_Operac...". The console output is as follows:

```
-----  
Antecessor e Sucessor  
-----  
Digite um numero: 8  
Analisando o numero 8, seu antecessor é 7 e seu sucessor é 9
```

## Exercício 02



A screenshot of a Windows console window. The title bar shows the file path: C:\Users\vinicius\OneDrive\Cesgranrio\LingProgIII\Aulas\Aula02\Ex03\_OperacoesAritimeticasConsole\bin\Debug\Ex03\_Operac... The console output is as follows:

```
-----  
Média do Aluno  
-----  
Digite o nome do aluno: Vinicius  
Digite a primeira nota: 7  
Digite a segunda nota: 8  
  
O aluno Vinicius, tirou as notas 7.0 e 8.0, e sua média é 7.5_
```

# Exercício 03



```
C:\Users\vinicius\OneDrive\Cesgranrio\LingProgIII\Aulas\Aula02\Ex04_OperacoesAritimeticasConsole\bin\Debug\Ex04_Operac...  
-----  
Preço do Produto  
-----  
Produto: Telefone  
Digite o valor do produto: 1000  
Digite o desconto (%): 12  
  
O produto Telefone custava R$R$ 1.000,00, com 12.00% de desconto, passa a custar  
R$ 880,00_
```

# Operadores Lógicos (Booleanos)

Operador Lógicos	Descrição
!	Negação
&&	AND
	OR
^	OR Exclusive

# Operadores de comparação e relacional

- ▶ Comparação >> "=", "!="
- ▶ Relacional >> ">", "<", ">=", "<="

# Operador Ternário

- ▶ `? >>` usado como operador ternário
- ▶ condição ? expressão1\_se\_true : expressão2\_se\_false
- ▶ Retorno Boolean;
- ▶ Atalho If...Else;



# Condições de controle IF (Simples)

- ▶ if ( condição )
  - ▶ {
    - ▶ comando 1;
    - ▶ comando 2;
  - ▶ }

# Condições de controle IF (Composta)

▶ If...else

▶ `if (nota >= 7)`

▶ `{`

▶ `resultado = "Aprovado";`

▶ `Console.WriteLine("Parabéns!");`

▶ `}`

`else`

`{`

▶ `resultado = "Reprovado";`

▶ `Console.WriteLine("Estude Mais!");`

▶ `}`

# Condições de controle IF (Encadeamentos)

```
▶ if (Condition1)
▶ {
▶     ▶ // Condition1 is true.
▶ }
▶ else if (Condition2)
▶ {
▶     ▶ // Condition1 is false and Condition2 is true.
▶ } else if (Condition3)
▶     ▶ Condition3 and Condition4 are true.
▶ {
▶ } else
▶ {
▶     ▶ // Condition1, Condition2, and Condition3 are false
▶ }
```

# Escopo e Inicialização

- ▶ Uma variável não pode ser usada se não for iniciada;
- ▶ Escopo de uma variável: é a região do programa onde a variável é válida, ou seja, onde ela pode ser referenciada;

# Condições de controle switch/case

- ▶ switch (variável ou valor)
- ▶ {
- ▶ case valor1:
- ▶     // código 1
- ▶ break;
- ▶ case valor2:
- ▶     // código 2
- ▶ break;
- ▶ }

# switch/case

```
▶ switch (variável ou valor)
▶ {
▶     case valor1:
▶     case valor2:
▶     case valor3:
▶         // código 1
▶         break;
▶     case valor4:
▶     case valor5:
▶     case valor6:
▶         // código 2
▶         break;
▶ }
```

# Exemplo

► `int caseSwitch = 1;`

```
switch (caseSwitch)
{
    case 1:
        Console.WriteLine("Case 1");
        break;
    case 2:
        Console.WriteLine("Case 2");
        break;
    default:
        Console.WriteLine("Default case");
        break;
}
```

# Estruturas de repetição

- ▶ For;
- ▶ While / Do While
- ▶ For each;



# For

```
▶ for (int i = 0; i <= 10; i++)  
▶     {  
▶         //instruções  
▶     }
```

# While / Do While

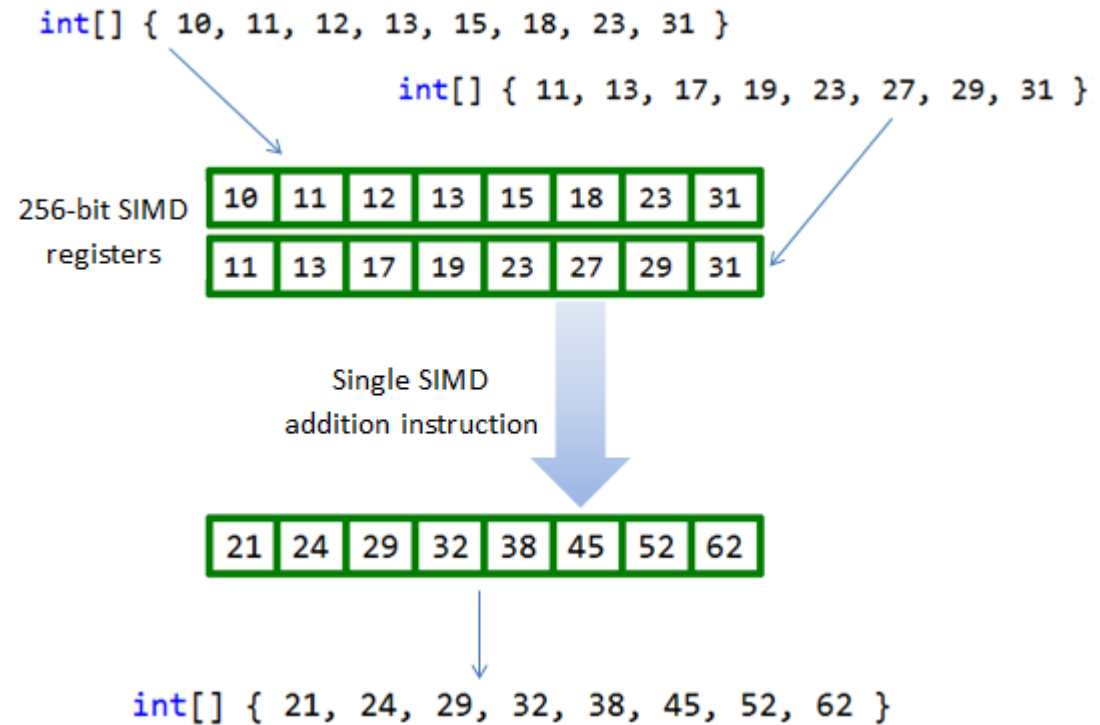
- ▶ Esta estrutura de repetição, garante que o bloco de instruções seja executado no mínimo uma vez, já que a condição que controla o laço é testada apenas no final do comando;
- ▶ A diferença entre o comando while e o do...while é justamente o local onde a condição que controla o laço é testada;
- ▶ No comando while a condição é testada antes do bloco de instruções, e caso a condição seja falsa a repetição não será executada;
- ▶ No do...while o bloco de comandos é executado pelo menos uma vez de forma obrigatória, independente do resultado da expressão lógica.

# Vetores

0	1	2	3
9	21	7	15

Ex:

```
int[] valores;  
float[] mediaAritimetica;  
  
valores = new int[10];
```



# Exemplos

- ▶ `valores = new int[4];`
- ▶ `valores[0] = 2;`
- ▶ `valores[1] = 10;`
- ▶ `valores[2] = 4;`
- ▶ `valores[3] = 1;`
- ▶ `String [] Nomes = New String [2];`
- ▶ `Nomes[0] = "João";`
- ▶ `Nomes[1] = "Marcos";`
- ▶ `Nomes[2] = "Maria";`

# Matrices

	0	1	2	3
0	"K"	"S"	"A"	"P"
1	"M"	"Q"	"C"	"Z"
2	"W"	"G"	"L"	"N"
3	"X"	"S"	"J"	"R"

```
int[,] numeros = new int[3,3];  
números[0 , 0] = 1;  
números[0 , 1] = 2;  
números[0 , 2] = 3;  
números[1 , 0] = 4;  
números[1 , 1] = 5;  
números[1 , 2] = 6;  
números[2 , 0] = 7;  
números[2 , 1] = 8;  
números[2 , 2] = 9;
```

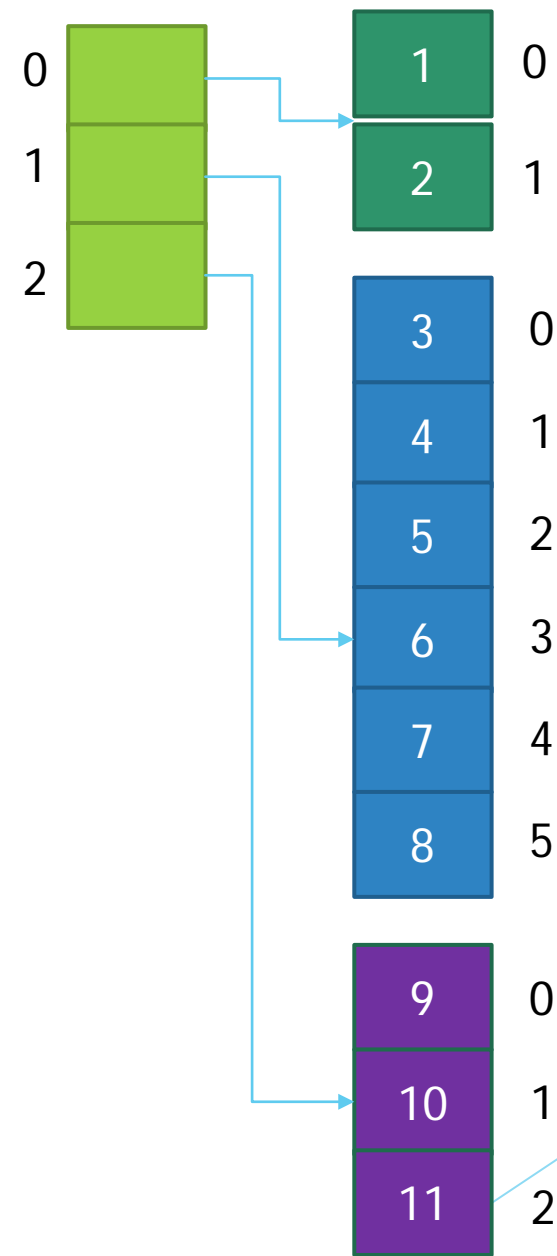
1	2	3
4	5	6
7	8	9

# Array

- ▶ `TIPO [] NomeDoArray = New TIPO [Tamanho do Array]`
- ▶ `TIPO [] NomeDoArray = New TIPO [Tamanho do Array, Tamanho do Array]`
- ▶ `float [ ] ValorIndice = new float [ 5 ]`
- ▶ `float [ , ] ValorIndice = new float [ 10 , 10 ];`
- ▶ `float [ ] ValorIndice = new float [ 5 ] { 1.25, 2, 3.23, 1.32, 5 };`
- ▶ `string [ , ] ElementoVetor = new string[3, 3] {{"ab", "ac", "bc"}, {"ab", "ac", "bc"}};`
- ▶ **jagged Array**
- ▶ `int [ ] [ ] MatrizDeInteiro = new int [ 2 ] [ ];`
  - ▶ `MatrizDeInteiro[ 0 ] = new int [ 5 ] {1,3,5,7,9};`
  - ▶ `MatrizDeInteiro[ 1 ] = new int [ 4 ] {2,4,6,8};`

# Jagged Array

- ▶ `int[][] jagged = new int[3][];`
- ▶ `jagged[0] = new int[2]{1,2};`
- ▶ `jagged[1] = new int[6]{3,4,5,6,7,8};`
- ▶ `jagged[2] = new int[3] { 9, 10, 11 };`



# Jagged Array - Exemplo

- ▶ Usuário Informa o número de Turmas;
- ▶ Usuário Informa o número de alunos (por turma);
  - ▶ Usuário insere o nome dos alunos (Conforme o Número de alunos inserido);



# Jagged Array – Continuação

- ▶ `Console.Write("Informe o número de turmas: ");`
- ▶ `int num_turmas = int.Parse(Console.ReadLine());`
- ▶ `string[][] turmas = new string[num_turmas][];`
- ▶ `for (int i = 0; i < num_turmas; i++)`
- ▶ `{`
  - ▶ `Console.Write("Informe o número de turmas: ");`
  - ▶ `Console.Write($"Informe a quantidade de alunos da turma {i}: ");`
  - ▶ `int num_alunos = int.Parse(Console.ReadLine());`
  - ▶ `turmas[i] = new string[num_alunos];`

