# C#
# Tipos de Variáveis e operadores

## Estruturas básicas

- Comentários
  - Identificar o responsável e a data de criação daquele código
  - Identificar todas as alterações feitas no código, seus responsáveis e datas de alteração
  - Identificar o objetivo do código
  - Explicar de maneira mais fácil a lógica do seu algoritmo
  - Tipos de comentários
    - /* */
    - //
    - /// comentário para documentação XML

## Estruturas básicas

- Blocos de comandos
  - Todos os comandos em C# tem que terminar em ponto-e-vírgula (;)
  - Delimitados por um bloco que tem início e fim, representados no C# pelos caracteres abre-chaves ({) que define o início do bloco e fecha-chaves (}) que define o fim do bloco

## Identação ou endentação ou indentação

- Termo aplicado ao código fonte de um programa de computador para indicar que os elementos hierarquicamente dispostos têm o mesmo avanço relativamente à posição (y,0) (linha, coluna).
- A identação tem um papel meramente "estético" (na maioria das linguagens) tornando a leitura do código fonte mais fácil (o que designamos por read-friendly).

## Definição de variáveis

- Variáveis são definidas como locais de armazenamento temporário de diferentes tipos como: números, palavras, datas e outros, que podem receber resultados de cálculos ou entrada de dados pelo usuário
- Prática muito utilizada hoje pelas empresas para um melhor entendimento das variáveis e também para facilitar a manutenção do seu código é no início de cada variável inserir letras minúsculas indicando o tipo da variável que você declarou no inicio do código

  Ex.: strNomeCliente (variável do tipo string)
  intIdadeCliente (variável do tipo int)

## Tipos de dados

int = Números inteiros (32 bits por padrão)
long = Números inteiros (64 bits por padrão)
float = Números de ponto flutuante (32 bits por padrão)
double = Números de ponto flutuante (64 bits por padrão)
string = Para seqüências de caracteres (16 bits por caractere)
char = Para somente um caractere (16 bits)
bool = Valor booleano que pode ser (true) ou (false) (1 bit)

| Type | Range | Size |
|---|---|---|
| sbyte | -128 to 127 | Signed 8-bit integer |
| byte | 0 to 255 | Unsigned 8-bit integer |
| char | U+0000 to U+ffff | Unicode 16-bit character |
| short | -32,768 to 32,767 | Signed 16-bit integer |
| ushort | 0 to 65,535 | Unsigned 16-bit integer |
| int | -2,147,483,648 to 2,147,483,647 | Signed 32-bit integer |
| uint | 0 to 4,294,967,295 | Unsigned 32-bit integer |
| long | -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 | Signed 64-bit integer |
| ulong | 0 to 18,446,744,073,709,551,615 | Unsigned 64-bit integer |

| Type | Approximate range | Precision |
|---|---|---|
| float | ±1.5e−45 to ±3.4e38 | 7 digits |
| double | ±5.0e−324 to ±1.7e308 | 15-16 digits |

# Declaração de variáveis

- Sintaxe

*TipoDeDado nomeDaVariavel [=valorInicial];*

- Exemplo

```
//declaração de uma variável do tipo inteira
// com o nome idade
int idade;
//declaração de uma variável do tipo inteira
// com o nome idade iniciada com valor 20
int idade = 20;
```

# Declaração de variáveis

- Duas variáveis do mesmo tipo

*double nota1, nota2, media;*
*nota1 = 8;*
*nota2 = 7;*
*media = (nota1 + nota2) / 2;*

Ou

*double nota1 = 8, nota2 = 7, media = (nota1 + nota2) / 2;*

# Operadores

**Multiplicative Operators**

| Expression | Description |
|---|---|
| * | Multiplication |
| / | Division |
| % | Remainder |

**Additive Operators**

| Expression | Description |
|---|---|
| x + y | Addition, string concatenation, delegate combination |
| x - y | Subtraction, delegate removal |

**Relational and Type Operators**

| Expression | Description |
|---|---|
| x < y | Less than |
| x > y | Greater than |
| x <= y | Less than or equal |
| x >= y | Greater than or equal |
| x is T | Return true if x is a T, false otherwise |
| x as T | Return x typed as T, or null if x is not a T |

**Equality Operators**

| Expression | Description |
|---|---|
| x == y | Equal |
| x != y | Not equal |

**Logical, Conditional, and Null Operators**

| Category | Expression | Description |
|---|---|---|
| Logical AND | x & y | Integer bitwise AND, Boolean logical AND |
| Logical XOR | x ^ y | Integer bitwise XOR, boolean logical XOR |
| Logical OR | x \| y | Integer bitwise OR, boolean logical OR |
| Conditional AND | x && y | Evaluates y only if x is true |
| Conditional OR | x \|\| y | Evaluates y only if x is false |
| Null coalescing | X ?? y | Evaluates to y if x is null, to x otherwise |
| Conditional | x ?: y : z | Evaluates to y if x is true, z if x is false |

# Conversão de Tipos

- É possível converter qualquer tipo primitivo para string mas nem sempre o contrário é possível;
- Um string que representa um número pode ser convertido para um tipo numérico, caso não represente, a conversão gerará uma exceção;
- Um número pode ser convertido para um número com mais bits
  - Ex.: byte (8 bits) para int (32 bits)

## Conversão de Tipos

- É possível converter automaticamente os tipos:
  - De byte, short, int, long, float, double, char, bool para string
  - De byte para short, de short para int, de int para float e de float para double

## Conversão de Tipos

- Mas pode ocorrer problemas na conversão dos tipos:
  - De double para float, long para int, de int para short, de short para byte
  - De uma string para um tipo numérico
- Nestes casos pode-se usar a classe **Convert**

## Exemplos

```
byte numero8bits = Convert.ToByte(“123”);

int numero32bits = Convert.ToInt32(“34132”);

double numeroReal64bits =
   Convert.ToDouble(“123123.12455”);

string palavra1 = Convert.ToString(123);
string palavra2 = 123.ToString();
```

## Exemplo: Média de três notas

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication1
{
   class Program
   {
      static void Main(string[] args)
      {
         //Declaração das variáveis
         string aluno;
         double nota1, nota2, nota3, media;
```

```
//Lendo as informações do usuário
        Console.Write("Informe a 1a nota: ");
        nota1 = Convert.ToDouble(Console.ReadLine());

        Console.Write("Informe a 2a nota: ");
        nota2 = Convert.ToDouble(Console.ReadLine());

        Console.Write("Informe a 3a nota: ");
        nota3 = Convert.ToDouble(Console.ReadLine());

//Processamento
        media = (nota1 + nota2 + nota3) / 3;

        //Saida
        Console.WriteLine("O aluno tirou as notas {0}, {1} e {2}", nota1, nota2, nota3);

        Console.WriteLine("Sua média é {0}", media);

        Console.ReadLine();
    }
  }
}
```

# Exemplo: Dias vividos

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace DiasVividos
{
    class Program
    {
        static void Main(string[] args)
        {
            //Declaração das variáveis
            string nome;
            int idade, dias_vividos;
```

```
//Lendo as informações do usuário
            Console.Write("Informa o seu nome: ");
            nome = Console.ReadLine();

            Console.Write("Informe a sua idade: ");
            idade = Convert.ToInt32(Console.ReadLine());

            //Processamento
            dias_vividos = idade * 365;

            //Saida
            Console.WriteLine("{0}, você tem {1} dias vividos", nome,
    dias_vividos);

            Console.ReadLine();
        }
    }
}
```

## Exercícios

- Faça um programa que peça duas notas para um aluno, os pesos correspondentes a cada nota (ex.: 6 e 4). Calcule e mostre a média ponderada dessas notas.
- Faça um programa que peça o salário de um funcionário, calcule e mostre o novo salário sabendo-se que este teve um aumento de 10%.

- Faça um programa que, informados três valores inteiros, calcule e exiba sua média.
- Faça um programa que converta uma medida de temperatura dada em Celsius (C) para Fahrenheit (F).

  F=9/5 *C +32