



Programação Orientada a Objetos

Unidade 1



Declarando e usando variáveis

```
//declara a idade
int idade;
int idade;
idadeNoAr
idade = 15;

// imprime a idade
Console.WriteLine(idade);
```

```
//gera a idade no ano seguinte
int idadeNoAnoQueVem;
idadeNoAnoQueVem = idade + 1;
```





Usando operadores

```
int quatro = 2 + 2;
int tres = 5 - 2;
int oito = 4 * 2;
 int dezesseis = 64 / 4;
 //5 dividido por 2 dá 2 e tem resto 1;
 int um = 5 % 2;
 // o operador % pega o resto da divisão inteira
```





Tipos primitivos

```
int i = 5; // i recebe uma cópia do valor 5
int j = i; // j recebe uma cópia do valor de i
int j = i; // i vira 6, j continua 5
i = i + 1; // i vira 6, j
```





1.3 Exercícios: Varáveis e tipos primitivos

- 1) Na empresa onde trabalhamos, há tabelas com o quanto foi gasto em cada mês. Para fechar o balanço do primeiro trimestre, precisamos somar o gasto total. Sabendo que, em Janeiro, foram gastos 15000 reais, em Fevereiro, 23000 reais e em Março, 17000 reais, faça um programa que calcule e imprima o gasto total no trimestre. Siga esses passos:
- a) Crie uma classe chamada BalancoTrimestral com um bloco Main, como nos exemplos anteriores;
- b) Dentro do Main (o miolo do programa), declare uma variável inteira chamada gastosJaneiro e inicialize-a com 15000;
- c) Crie também as variáveis gastosFevereiro e gastosMarco, inicializando-as com 23000 e 17000, respectivamente, utilize uma linha para cada declaração;
- d) Crie uma variável chamada gastosTrimestre e inicialize-a com a soma das outras 3 variáveis:

int gastosTrimestre = gastosJaneiro + gastosFevereiro + gastosMarco;

e) Imprima a variável gastosTrimestre.





Casting

```
double d = 3.1415;
int i = d; // não compila
```

```
int i = 5;
double d2 = i;
```

// compila ??????





Casting

```
long x = 10000;
int i = (int)x;
```





Casting

						long	float	double	
	PARA:	byte	short	char	int		Impl.	Impl.	11
D	E:		Impl.	(char)	Impl.	Impl.	Impl.	Impl.	
_	yte	(byte)		(char)	Impl.	Impl.	Impl.	Impl.	1
-	hort	(byte)	(short)		Impl.	Impl.	Impl.	Impl.	\dashv
	char int	(byte)	(short)	((int)		Impl.	Impl.	7
1	long	(byte)	T (1 +)	1 1 - 1	(int)	1	(Float		1
1 1	float	(byte)	1 (-hart		(: m+)	(long) (Hoat	.7	
	double	(byte)	(311012	,					





O if e o else

```
int idade = 15;
if (idade < 18)

{
        Console.WriteLine("Não pode entrar");
}
else
{
        Console.WriteLine("Pode entrar");
}</pre>
```





O if e o else

```
int mes = 1;
if (mes == 1)
{
        Console.WriteLine("Você deveria estar de férias");
}
```

```
int idade = 15;
bool amigoDoDono = true;
bool amigoDoDono = true;
if (idade < 18 & !amigoDoDono)

if (console.WriteLine("Não pode entrar");
}
else
{
    Console.WriteLine("Pode entrar");
}</pre>
```





O While

```
int idade = 15;
while (idade < 18)
{
    Console.WriteLine(idade);
    idade = idade + 1;
}</pre>
```





O for

```
for (inicializacao; condicao; incremento)
{
    codigo;
}

Um exemplo é o a seguir:

for (int i = 0; i < 10; i = i + 1)
{
    console.WriteLine("olá!");
}</pre>
```





Controlando loops





Escopo das variáveis

```
//aqui a variável i não existe
int i = 5;
// a partir daqui ela existe
while (condicao)
{
    // o i ainda vale aqui
    int j = 7;
    // o j passa a existir
}
// aqui o j não existe mais, mas o i continua a valer
j = 8;
```

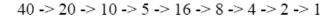




1.13 - Exercícios: Fixação de sintaxe

Não copie e cole de um exercício já existente! Aproveite para praticar.

- 1) Imprima todos os números de 150 a 300.
- 2) Imprima a soma de 1 até 1000.
- Imprima todos os múltiplos de 3, entre 1 e 100.
- 4) (opcional) Imprima os primeiros números da série de Fibonacci até passar de 100. A série de Fibonacci é a seguinte: 0, 1, 1, 2, 3, 5, 8, 13, 21, etc... Para calculá-la, o primeiro e segundo elementos valem 1, daí por diante, o n-ésimo elemento vale o (n-1)ésimo elemento somando ao (n-2)-ésimo elemento (ex: 8 = 5 + 3).
- 5) (opcional) Escreva um programa que, dada uma variável x (com valor 180, por exemplo), temos y de acordo com a seguinte regra:
 - se x é par, y = x / 2
 - se x é impar, y = 3 * x + 1
 - imprime y
 - O programa deve então jogar o valor de y em x e continuar até que y tenha o valor final de 1. Por exemplo, para x = 13, a saída será:







1.12 - Um pouco mais...

- Vimos apenas os comandos mais usados para controle de fluxo. O C# ainda possui o do..while e o switch. Pesquise sobre eles e diga quando é interessante usar cada um deles.
 - 2) O que acontece se você tentar dividir um número inteiro por 0? E por 0.0?
 - 3) Existem outros operadores, como o %, <<, >>. Descubra para que servem.
- 4) Além dos operadores de incremento, existem os de decremento, como --i e i--. Além desses, você pode usar instruções do tipo i += x e i -= x, o que essas instruções fazem? Teste.







Programação Orientada a Objetos

Apresentação da IDE Visual Studio - Revisão