LINGUAGEM DE PROGRAMAÇÃO

Orientada a objeto

Roteiro da aula

- Aplicação Console
- Instruções e Expressões
- Variáveis
- Tipos de Dados
- Conversão de Dados
- Operadores Aritméticos, Relacionais e Lógicos
- Formatação Numérica

Aplicação Console

- Aplicação que roda atravez de uma linha de comando, no prompt do DOS.
- Crie um novo projeto do tipo – Console Application
- A tela inicial já é a de código. Não tem interface grafica.
- Adicione o codigo a seguir dentro do metodo Main:

```
string curso = “C#”;
Console.WriteLine(curso);
```

- Execute com CTRL+F5

Codigo:

- Para comentario usamos // para comentar uma frase ou para comentar um paragrafo /* no inicio e */ no final
- Metodo Main → onde cria os objetos e executa os outros metodos.
- Declaração de variaveis → tipo nome = valor

  Exemplo: string nome = “Carlos”;
- Entrada e Saida → WriteLine é um metodo de saida da classe Console. Exibe no console o valor do parâmetro seguido por uma nova linha.

  → ReadLine é um metodo de entrada da classe console Exemplo: Variavel = Console.ReadLine()

  → ReadKey metodo para pausas a tela

Exercite

1) Crie um projeto Console com o texto: “Digite seu nome”, armazene em uma variavel e exiba na tela : “O nome é: ” com o nome digitado.

Exercite

2) Crie um projeto console com 2 variaveis: idade do tipo int e nome do tipo string e atribua um valor a cada uma delas. Exiba na tela:

As variaveis são:

nome = nome

idade = idade

Instruções, Expressões e operadores

- O codigo C# consiste de instruções abrangendo palavras chaves C#, expressões e operadores
- A instrução deve ser criada necessariamente dentro do metodo
- Uma declaração de variável pode ser feito dentro da classe

```
public partial class frmCadastro : Form
{
   String status = "";
   private void cmdIncluir_Click(object sender, EventArgs e)
   {
      Incluir();
   }

   private void Incluir()
   {
      status = "Empresa: " + txtEmpresa.Text + " incluída.";
      lblStatus.Text = status;
   }

}
```

Variáveis

- As variáveis locais devem ser inicializadas antes de serem usadas:

  int x = 0; ou int x = new int();

  valor padrão C# para int =0

- Exemplos de valores padrão:

  bool → False

  char → ‘ \0 ’

  decimal → 0.0M

  double → 0.0

  float → 0.0F

  int → 0

Visibilidade da variavel

```
 9  namespace MeuProjeto
10  {
11      public partial class frmCadastro : Form
12      {
13          public frmCadastro()...
17
18          private void cmdIncluir_Click(object sender, EventArgs e)
19          {
20              Incluir();
21              status = "Erro";
22          }
23
24          private void Incluir()
25          {
26              String status;
27              status = "Empresa: " + txtEmpresa.Text + " incluída.";
28              lblStatus.Text = status;
29          }
30
31      }
32  }
```

Exercite Windows Form

- Exercicio 1

Principais Tipos de Dados no C#

Tipo de Dado	Descrição	Tamanho (bits)	Intervalo	Exemplo de Utilização
short	nºs inteiros	16	Entre -32768 e 32767	
int	nºs inteiros	32	Entre -2.147.483.648 a 2.147.483.648	int cont cont = 10;
long	nºs inteiros (intervalo maior)	64	Entre -9.223.372.036.854.775.808 a 9.223.372.036.854.775.808	long dist dist = 42L;
float	nºs de ponto flutuante	32	Entre -3.4×10^{38} a $+ 3.4 \times 10^{38}$	float val val = 0.42F;
double	nºs de ponto flutuante de dupla precisão	64	Entre -1.7×10^{308} a $+1.7 \times 10^{308}$	double y y = 0.42;
decimal	Valores monetários	128	28 números significativos	decimal tot tot = 0.42M;
string	Sequência de caracteres	16 bits por caractere	Não aplicável	string nome nome = "zé";
char	Caractere único	16	De 0 a 32.768 unicode.	char sexo sexo = "M";
bool	Booleano	8	Verdadeiro ou falso	bool resp resp = false;

Principais Tipos de Dados no C#

- BOOL →O tipo bool armazena os valores true e false. As expressões lógicas como if e while esperam sempre uma expressão do tipo bool

```
1  ...
2  bool EValido = true;
3  if (EValido)
4  {
5          Console.WriteLine("É valido");
6  }
7  else
8  {
9          Console.WriteLine("Nada feito");
10 }
11 ...
```

Principais Tipos de Dados no C#

Tipo Long - Float - Decimal

- Estes tipos de variáveis deverão ser iniciadas, respectivamente, com o sufixo **L, F** e **M**.
- Caso você se esqueça de acrescentar o sufixo F a uma variável float, o compilador irá tratar o valor como double
- O mesmo acontece com o decimal.
- Já a variável do tipo long que será tratada como int
- Exemplo:

```
long x = 1000L;
float valor = 5.87F;
decimal aluguel = 650.00M;
```

Principais Tipos de Dados no C#

- **Caractere (por valor) → CHAR**
- O que diferencia efetivamente o texto atribuído a um tipo string, é que a atribuição de um caracterer é feita envolvendo o texto por aspas simples (‘) enquanto para a string o texto é envolvido por aspas duplas (“).

Conversão

- Tudo o que for digitado na tela é do tipo string.
- Quando precisamos armazenar ou manipular esse dado devemos fazer a conversão para int ou double
- Para visualizar novamente o resultado na tela esse valor deve necessariamente ser convertido para String.

Conversão de dados para calculo

```
string nome, idade1, idade2, salario1, salario2
...

double somaSalario = double.Parse(salario1) +
                                    double.Parse(salario2);
int idade = int.Parse(idade1) + int.Parse(idade2);

...

ConsoleWriteLine("SomaSalario="+somasalario.ToString());
```

Conversão de dados para calculo

```
string nome, idade1, idade2, salario1, salario2
...

double somaSalario = Convert.ToDouble(salario1) +
                                    Convert.ToDouble (salario2);
int idade = Convert.ToInt(idade1) + Convert.ToInt (idade2);

...

ConsoleWriteLine("SomaSalario="+somasalario.ToString());
```

Operadores Aritméticos

Sendo: int a, b, x; a = 10; b = 5;

Operador	Operação	Exemplo	Resultado
+	Adição ou soma	x = a + b;	x = 15
-	Subtração	x = a – b;	x = 5
*	Multiplicação	x = a * b;	x = 50
/	Divisão	x = a / b;	x = 2
%	Resto ou divisão modular	x = a % b;	x = 0
++	Incremento	a++;	a = 11
--	Decremento	b--;	b = 4
+=	Acumulador mais igual	a += 2;	a = 13
-=	Acumulador menos igual	b -= 2;	b = 2
*=	Acumulador vezes igual	a *= 2;	a = 26
/=	Acumulador divisão igual	b /= 2;	b = 1
%=	Acumulador módulo igual	b %=2;	b = 1
()	Precedência	x = (a + b) / 5;	x = 5

Operadores Relacionais

Sendo: int a, b, x; a = 10; b = 5;

Operador	Operação	Exemplo	Resultado
<	Menor que	a < b	*false*
<=	Menor igual a	a <= b	*false*
>	Maior que	a > b	*true*
>=	Maior igual a	a >= b	*true*
==	Comparação	a == b	*false*
		a == a	*frue*
!=	Não igual	a != b	*true*
		a != a	*false*
=	Atribuição	a = b	*a = 5*
!	Não	(!(a < b))	*true*

Operadores Lógicos

- São utilizados para realizar comparações ou testar condições entre duas ou mais expressões relacionais.
- Existem dois operadores lógicos:

  operador aditivo && (And = e)

  operador alternativo || (Or = ou).
- Em cada expressão lógica resultará um resultado booleano

Exercite

- Exercicio 2 e 3

Formatação Numerica

- Usando String.Format

```
double x = 65.550;
lblStatus.Text = String.Format ("{0:C}",x);
```

Caractere	Descrição	Exemplos	Saída
C ou c	Moeda	Console.Write("{0:C}", 2.5); Console.Write("{0:C}", -2,5);	R$ 2,50 (R$ 2,50)
D ou d	Decimal	Console.Write("{0:D5}", 25);	00025
E ou e	Científico	Console.Write("{0:E}", 250000);	2.500000E + 005
F ou f	Ponto fixo	Console.Write("{0:F2}", 25); Console.Write("{0:F0}", 25);	25.00 25
G ou g	Gerais	Console.Write("{0:G}", 2.5);	2.5
N ou n	Número	Console.Write("{0:N}", 2500000);	2,500,000.00

Exercite

Exercicio 4.