

Apresentação

Curso:

Programação .NET I

Aula 02:

Objetivos:

Esquema de diretórios;

Dados primitivos;

Operadores e Precedência;

Estruturas de Controle (if, switch);

Esquema de diretórios

Vamos aprender como estão organizados os arquivos da sua solução (Solution).

Abra o Visual Studio 2012 e crie uma nova *Solution* com as seguintes características:

Tipo: C#; Windows; Console Application.

Name: “dados_primitivos”

Location: “c:\users\[username]\documents\visual studio 2012\Projects”

Solution Name: “Nome_Sobrenome-Aula02”



Esquema de diretórios

Vamos observar os arquivos criados pelo Visual Studio 2012.

Abram o diretório de “Documentos” do usuário ativo. (Windows + E) → Documentos

Dados primitivos

Os dados primitivos são valores classificados à um determinado tipo de dado básico da linguagem de programação.

No nosso caso, teremos os seguintes tipos:

Dados primitivos

Tipo	Descrição	Tamanho (em bits)	Intervalo	Exemplo de utilização
int	Números inteiros	32	-2,147,483,648 até +2,147,483,647	int cont; cont = 10;
long	Números inteiros (intervalo maior)	64	-9,223,372,039,854,775,808 até +9,223,372,039,854,775,807	long tamanho; tamanho = 10L;
float	Números de ponto flutuante	32	(+ ou -) 3.4×10^{38}	float dizima; dizima = 0.78F;
double	Números de ponto flutuante de precisão dupla	64	(+ ou -) 1.7×10^{308}	double dizima2; dizima2 = 0.78;
decimal	Valores monetários	128	28 números significativos	decimal preco; preco = 0.83M;
string	Sequência de caracteres	16 bits por caractere	NA	string palavra; palavra = "Teste";
char	Caractere único	16	0 até 32,768	char letra; letra = '3';
bool	Booleano	8	verdadeiro ou falso	bool teste; teste = false;

* Utilização de sufixos para garantir a alocação específica do dado.

Dados primitivos

Boas prática para variáveis:

- Não usar “underline” (sublinhado) “_”;
- Não usar identificadores como:
 “guardaValorTeste” e “guardaValorteste”.
- Começar o nome com letra minúscula
(Notação CamelCase);

Dados primitivos

Exercício 1: (Application name: dados_primitivos)

Escreva o código abaixo no método Main da classe Program .

```
string palavra;  
Console.WriteLine(palavra);
```

Dados primitivos

Dica:

O C# não permite a utilização de variáveis não inicializadas!

```
string palavra;  
Console.WriteLine(palavra);
```

- Vai gerar um erro em tempo de compilação.
(*Error 1 Use of unassigned local variable 'palavra'*)

Operadores e precedência

Assim como em outras linguagens, os operadores da linguagem C# possuem ordem de precedência, associação e funcionalidade.

A *precedência* ou prioridade de um operador é quem controla a ordem da qual os operadores de expressão são avaliados (executados).

A *associação* é quem controla para qual lado da expressão o operador irá executar a sua função.

Operadores e precedência

Categoria	Operadores	Descrição	Associação
Primário	() ++ --	Substitui precedência Sufixo/Prefixo de incremento Sufixo/Prefixo de decremento	Esquerda
Unário	! + -	Negação Lógica Identidade Negação	Esquerda
Multiplicativo	* / %	Multiplicação Divisão Resto da divisão	Esquerda
Aditivo	+ -	Adição Subtração	Esquerda
Relacional	< <= > >=	Menor que Menor ou igual que Maior que Maior ou igual que	Esquerda
Igualdade	== !=	Igualdade Diferente de	Esquerda
Condicional AND	&&	AND lógico	Esquerda
Condicional OR		OR lógico	Esquerda
Atribuição	=	Atribuição	Direita

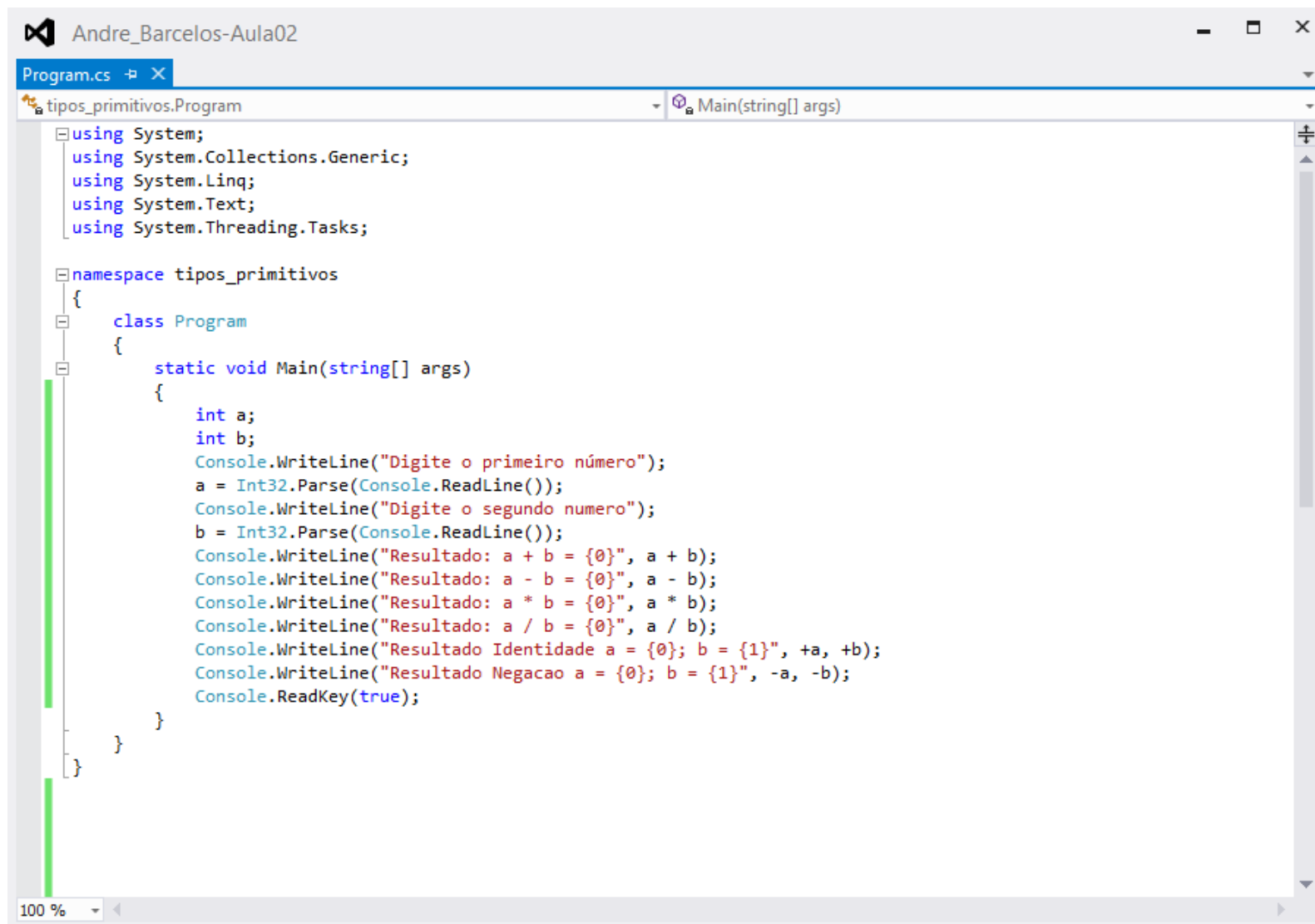
Exercício

Exercício 2: (Application name: “operadores primitivos”)

Faça um programa (Console) que leia dois números (inteiros) e apresente os resultados das operações com os operadores: Aditivos (soma e subtração), Multiplicativos (multiplicação e divisão), de Identidade e de Negação.

PS: Utilize o método *Parse* da estrutura *System.Int32*

Resposta do exercício



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

namespace tipos_primitivos
{
    class Program
    {
        static void Main(string[] args)
        {
            int a;
            int b;
            Console.WriteLine("Digite o primeiro número");
            a = Int32.Parse(Console.ReadLine());
            Console.WriteLine("Digite o segundo numero");
            b = Int32.Parse(Console.ReadLine());
            Console.WriteLine("Resultado: a + b = {0}", a + b);
            Console.WriteLine("Resultado: a - b = {0}", a - b);
            Console.WriteLine("Resultado: a * b = {0}", a * b);
            Console.WriteLine("Resultado: a / b = {0}", a / b);
            Console.WriteLine("Resultado Identidade a = {0}; b = {1}", +a, +b);
            Console.WriteLine("Resultado Negacao a = {0}; b = {1}", -a, -b);
            Console.ReadKey(true);
        }
    }
}
```

Estrutura de controle (if)

```
if (EXPRESSÃO LÓGICA)
{
    BLOCO DE INSTRUÇÕES "TRUE"
}
else
{
    BLOCO DE INSTRUÇÕES "FALSE"
}
```

```
if (EXPRESSÃO LÓGICA)
    LINHA DE INSTRUÇÃO "TRUE";
else
    LINHA DE INSTRUÇÃO "FALSE";
```



Operadores e estrutura *if*

Operador	Significado	Exemplo	Resultado se <code>valor</code> for 42
<code>==</code>	Igual a	<code>valor == 200</code>	false
<code>!=</code>	Diferente de	<code>valor != 1</code>	true
<code><</code>	Menor que	<code>valor < 83</code>	true
<code><=</code>	Menor ou igual a	<code>valor <= 20</code>	false
<code>></code>	Maior que	<code>valor > 10</code>	true
<code>>=</code>	Maior ou igual a	<code>valor >= 83</code>	false

Operadores e estrutura *if*

Short circuiting:

Os operadores `&&` e `||` possuem um recurso chamado de “short circuiting” (curto-circuito). Em algumas situações não é necessário calcular os dois termos (operandos) de uma expressão.

Exemplo 1:

```
(valor >= 20) && (valor <= 100)
```

Exemplo 2:

```
(valor < 10) || (valor > 100)
```

Estrutura de controle (if)

Dicas para usar a estrutura if:

```
int tempo;  
...  
if (tempo = 30) //erro de compilação! (acontecia em C)  
...  
if (tempo == 30) //ok!
```

----- //

```
bool teste;  
...  
if (teste == true) //não é comum, mas pode!  
...  
if (teste) //boa prática
```


Exercício

Exercício 3a: (Application name: “estrutura_if”)

Adicione uma nova application à Solution para fazer um programa (Console) que leia um número (inteiro) e dê uma mensagem de console com o nome do dia da semana correspondente. Sendo:

1 = Domingo ... 7 = Sábado

PS: Utilize o método *Parse* da estrutura *System.Int32*.

Resposta do exercício

```
Andre_Barcelos-Aula02 - Program.cs*
Program.cs*
Estruturas_de_controle.Program
Main(string[] args)

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Estruturas_de_controle
{
    class Program
    {
        static void Main(string[] args)
        {
            string nomeDiaSemana = "";

            Console.WriteLine("Programa de estrutura if");
            Console.WriteLine("Digite um numero inteiro entre 1 e 7");
            int diaSemana = Int32.Parse(Console.ReadLine());

            if (diaSemana == 1)
                nomeDiaSemana = "Domingo";
            else if (diaSemana == 2)
                nomeDiaSemana = "Segunda-feira";
            else if (diaSemana == 3)
                nomeDiaSemana = "Terça-feira";
            else if (diaSemana == 4)
                nomeDiaSemana = "Quarta-feira";
            else if (diaSemana == 5)
                nomeDiaSemana = "Quinta-feira";
            else if (diaSemana == 6)
                nomeDiaSemana = "Sexta-feira";
            else if (diaSemana == 7)
                nomeDiaSemana = "Sábado";
            else
                nomeDiaSemana = "Desconhecido";

            Console.WriteLine("Dia escolhido: " + nomeDiaSemana);
            Console.ReadKey(true);
        }
    }
}
```

Estrutura de controle (switch)

```
switch (VARIÁVEL TESTADA)
{
    case VALOR1:
        BLOCO DE INSTRUÇÕES
        break;
    case VALOR2:
        BLOCO DE INSTRUÇÕES
        break;
    case VALOR3:
        BLOCO DE INSTRUÇÕES
        break;
    [default:
        BLOCO DE INSTRUÇÕES //por convenção fica por último
    break;]
}
```

Estrutura de controle (switch)

Regras para usar a switch:

- Com tipos de dados primitivos (int ou string).
- Quando as expressões testadas são constantes, como 1, 2... Ou “Domingo”, “Segunda”...
- Os rótulos *case* devem ser únicos. Ou seja, dois rótulos *case* não podem ter o mesmo valor.
- Respeitar o “no fall-through”.



Exercício

Exercício 3b: (Application name: “estrutura_switch”)

Adicione uma nova application à Solution para fazer um programa (Console) que leia um número (inteiro) e dê uma mensagem de console com o nome do dia da semana correspondente. Sendo:

1 = Domingo ... 7 = Sábado

PS: Utilize o método *Parse* da estrutura *System.Int32*.

Resposta do exercício

```
switch (diaSemana)
{
    case 1:
        nomeDiaSemana = "Domingo";
        break;
    case 2:
        nomeDiaSemana = "Segunda-feira";
        break;
    case 3:
        nomeDiaSemana = "Terça-feira";
        break;
    case 4:
        nomeDiaSemana = "Quarta-feira";
        break;
    case 5:
        nomeDiaSemana = "Quinta-feira";
        break;
    case 6:
        nomeDiaSemana = "Sexta-feira";
        break;
    case 7:
        nomeDiaSemana = "Sábado";
        break;
    default:
        nomeDiaSemana = "Desconhecido";
        break;
}
```

Resposta do exercício

```
class Program
{
    static void Main(string[] args)
    {
        string nomeDiaSemana = "";

        Console.WriteLine("Programa de estrutura switch");
        Console.WriteLine("Digite um numero inteiro entre 1 e 7");
        int diaSemana = Int32.Parse(Console.ReadLine());

        //estrutura switch

        Console.WriteLine("Dia escolhido: " + nomeDiaSemana);
        Console.ReadKey(true);
    }
}
```

Resumo de aula

- Nesta aula vimos:
 - Estruturas e diretórios de arquivos do Visual Studio 2012
 - Operadores e precedência
 - Tipos primitivos
 - Exercício prático
 - Estruturas de controles
 - if
 - Exercício prático
 - switch
 - Exercício prático

Resumo de aula

- Mandem as anotações de aula para a tarefa no moodle.
- A tarefa “Trabalho – 02” é correspondente ao trabalho sobre Frameworks de testes em .NET