EAD
UNISANTA

Programação Orientada a Objetos

Ms. Eng. Claudio Ferreira de Carvalho

AULA 05

Operadores aritméticos e conversão de variáveis – Comandos Read e ReadLine

Operadores Aritméticos

✓ São símbolos que representam as operações aritméticas

Operação	Função	Significado	Sintaxe em C#
+	Adição	Soma valores	a + b
-	Subtração	Subtrai valores	a – b
*	Produto	Multiplica valores	a * b
/	Divisão	Divide valores	a / b
%	Resto	Resto da divisão	a%b
++	Incremento	Acrescenta 1 ao valor	a++ ou ++a
--	Decremento	Subtrai 1 ao valor	a-- ou --a

Caso se deseje a parte inteira, basta declarar os valores como inteiros

Se o incremento for colocado após a variável (a++), o incremento será feito depois de usar o valor da variável na expressão.

Se o incremento for colocado antes da variável (++a), o incremento será feito antes do uso do valor na expressão.

Precedência de Operações

- ✓ São símbolos que representam as operações aritméticas

Prioridade	Operadores
1	Parênteses
2	Potenciação e Radiciação
3	% * /
4	+ -

Exemplos importantes

8 + 4/2 = 10 (Primeiro divide 4 por 2 e posteriormente soma com 8

(8 + 4)/2 = 6 (Primeiro soma o que está no parêntese 8 + 4 = 12 e posteriormente divide por 2

Comando Read

- ✓ Objetivo
  - ✓ Interrompe o fluxo do programa e aguarda o usuário digitar algo e encerrar com <Enter>
  - ✓ Ao receber a digitação, atribui o código decimal do primeiro caracter digitado (conforme tabela ASCII) à variável cujo indicador foi colocando antes do comando Read
- ✓ Sintaxe

```
foidigitado = Console.Read();
```

A vantagem é que se for digitado mais de um caracter ele só recebe o primeiro

Tabela ASCII

Caracter	Dec	Oct	Hex
(nul)	0	0000	0x00
(soh)	1	0001	0x01
(stx)	2	0002	0x02
(etx)	3	0003	0x03
(eot)	4	0004	0x04
(enq)	5	0005	0x05
(ack)	6	0006	0x06
(bel)	7	0007	0x07
(bs)	8	0010	0x08
(ht)	9	0011	0x09
(nl)	10	0012	0x0a
(vt)	11	0013	0x0b
(np)	12	0014	0x0c
(cr)	13	0015	0x0d
(so)	14	0016	0x0e
(si)	15	0017	0x0f
⋮	⋮	⋮	⋮
²	253	0375	0xfd
_	254	0376	0xfe
	255	0377	0xff

Comando Read – Convertendo valor numérico em caracter

- Objetivo
  - A instrução Convert.ToChar, converte um valor numérico no caracter ASCII correspondente.
- Sintaxe

```
referenciaAscii = Convert.ToChar(codigoDecimal);
```

Comando Read – Convertendo valor numérico em caracter

- ✓ Objetiv
  - ✓ A ins
- ✓ Sintaxe
  referenci
- ✓ Exemplo

```
1.  namespace Aula05_Ex01
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // declaração de variáveis
8.              int codDec;
9.              char referencia;
10.
11.             // Recebe valor
12.             Console.Write("Digite um caracter qualquer: ");
13.             codDec = Console.Read();
14.
15.             // Converte o valor recebido para caracter
16.             referencia = Convert.ToChar(codDec);
17.
18.             Console.WriteLine("O código decimal referente ao caracter digitado é: " + codDec);
19.             Console.WriteLine("O caracter digitado foi " + referencia);
20.
21.             // Mantem a tela aguardando a digitação de uma tecla
22.             Console.ReadKey();
23.         }
24.     }
25. }
```

Execução

```
C:\D\Aulas_Atuais\Curso_Programacao_Orientada_Objetos_2024\Projetos_POO\Au
Digite um caracter qualquer: X
O código decimal referente ao caracter digitado é: 88
O caracter digitado foi X
```

Comando ReadLine

- ✓ Objetivo
  - ✓ Interrompe o fluxo do programa e aguarda o usuário digitar algo e encerrar com <Enter>.
  - ✓ Ao receber a digitação, atribui o conteúdo digitado à variável cujo indicador foi colocado antes do comando ReadLine
- ✓ Sintaxe

```
foidigitado = Console.ReadLine();
```

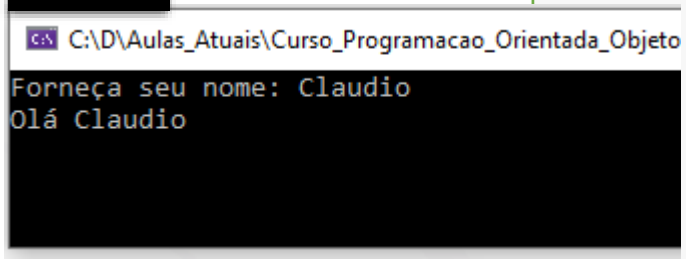
O valor digitado é sempre recebido como string, portanto, se for desejado valor numérico tem que ser convertido como será visto futuramente.

Comando ReadLine

- ✓ Objetivo
  - ✓ Interrompe
  - ✓ Ao receb
    comando
- ✓ Sintaxe
  foidigitado =
- ✓ Exemplo

```
1. namespace Aula05_Ex02
2. {
3.     internal class Program
4.     {
5.         static void Main(string[] args)
6.         {
7.             // Declara Variáveis
8.             string nome;
9.
10.            // Recebe o nome
11.            Console.Write("Forneça seu nome: ");
12.            nome = Console.ReadLine();
13.
14.            // Informa os valores
15.            Console.WriteLine("Olá {0}", nome);
16.
17.            // Mantem a tela aberta esperando a digitação de uma tecla
18.            Console.ReadKey();
19.        }
20.    }
21.}
```

Execução



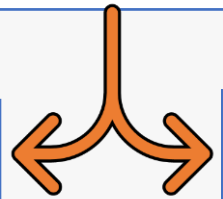
Comando ReadLine – Recebendo conteúdo tipo char

- ✓ Objetivo
  - ✓ O comando ReadLine nativo recebe conteúdos String.
  - ✓ Para converter em conteúdo tipo char precisa ser convertido com char.Parse
- ✓ Sintaxe

```
foidigitado = char.Parse(Console.ReadLine()); // converte o que foi digitado em char
```

Como o comando char só recebe um caracter, se for digitado mais de um caracter o programa apresentará um erro.

Para evitar este erro pode-se:



Ser utilizado o Comando Read, que so recebe o primeiro caracter digitado

Contar o número de caracteres digitados e evitar a atribuição co char.Parse, solicitando que o usuário digite corretamente.
Para fazer esta proteção, são necessários comandos que serão vistos futuramente neste curso.

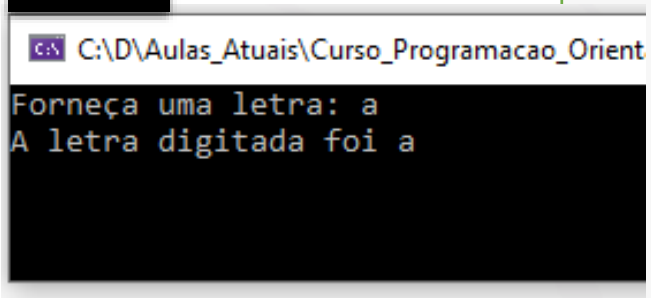
Comando ReadLine – Recebendo conteúdo tipo char

- ✓ Objetivo
  - ✓ O comand
  - ✓ Para conv
- ✓ Sintaxe

  foidigitado = c
- ✓ Exemplo

```
1. namespace Aula05_Ex03
2. {
3.     internal class Program
4.     {
5.         static void Main(string[] args)
6.         {
7.             // Declara Variáveis
8.             char letra;
9.
10.            // Recebe uma letra
11.            Console.Write("Forneça uma letra: ");
12.            letra = char.Parse(Console.ReadLine());
13.
14.            // Informa os valores
15.            Console.WriteLine("A letra digitada foi {0}", letra);
16.
17.            // Mantem a tela aberta esperando a digitação de uma tecla
18.            Console.ReadKey();
19.
20.        }
21.    }
22.}
```

Execução



Comando Read – Recebendo conteúdo tipo char

Observar que res é um valor numérico inteiro correspondente ao primeiro caracter digitado conforme tabela ASCII

Em seguida o valor numérico é convertido para char

```
1.  namespace Aula05_Ex04
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // Declara Variáveis
8.              int res;
9.              char resposta;
10.
11.             // Recebe digitação - com comando Read e convertre para decimal da ASCII
12.             Console.Write("Digite S para Sim ou N para Não. ");
13.             res = Console.Read();
14.             // Converte o valor recebido para caracter
15.             resposta = Convert.ToChar(res);
16.
17.             // O código ASCII do primeiro caracter digitado é:
18.             Console.WriteLine("Código ASCII di caracter digitado {0} ", res);
19.
20.             // Informa a primeira letra digitada
21.             Console.WriteLine("Sua resposta foi {0} ", resposta);
22.
23.             // Mantem a tela aberta esperando a digitação de uma tecla
24.             Console.ReadKey();
25.         }
26.     }
27. }
```

Execução

```
C:\D\Aulas_Atuais\Curso_Programacao_Orientada_Objetos_2024\
Digite S para Sim ou N para Não. Sim
Código ASCII di caracter digitado 83
Sua resposta foi S
```

ReadLine – como receber um número inteiro

- Objetivo
  - O comando ReadLine, recebe variáveis do tipo string
  - Caso a variável à qual se deseja atribuir um comando digitado seja um número inteiro (byte, sbyte, ushort, short, uint, int, ulong, long), é necessário converter com o commando Parse
- Sintaxe

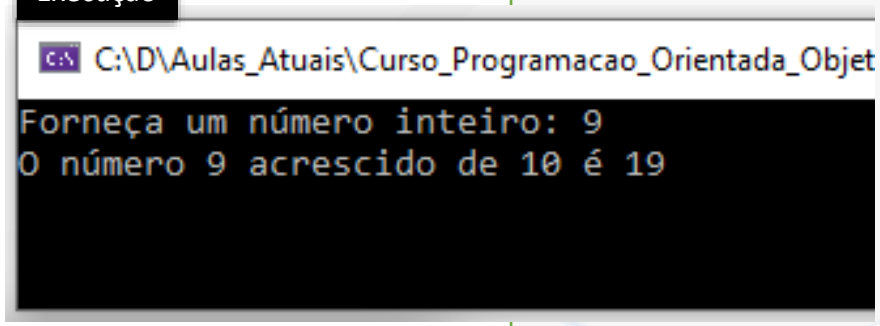
```
foidigitado = int.Parse(Console.ReadLine());
foidigitado = byte.Parse(Console.ReadLine());
foidigitado = short.Parse(Console.ReadLine());
foidigitado = long.Parse(Console.ReadLine());
      :       :       :
      :       :       :
      :       :       :
```

Caso o usuário digite caracteres ao invés de números o programa apresentará erro em tempo de execução.
Para resolver este problema pode-se utilizar a instrução int.TryParse(Console.Readline(), out variável).
Esta utilização necessita de uma interpretação do valor atribuído à variável, pois ele será zero (0) se o caracter fornecido não for numérico.
Como, os comandos necessários para este tratamento serão vistos somente a partir da próxima aula, estas implementações só serão feitas futuramente..

ReadLine – como receber um número inteiro do tipo int

```
1.  namespace Aula05_Ex05
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // Declara Variáveis inteira
8.              int ni, resInt;
9.
10.             // Recebe um valor
11.             Console.Write("Forneça um número inteiro: ");
12.             ni = int.Parse(Console.ReadLine());
13.
14.             // Atribui valor
15.             resInt = ni + 10;
16.
17.             // Informa os valores
18.             Console.WriteLine("O número {0} acrescido de 10 é {1}", ni, resInt);
19.
20.             // Mantem a tela aberta esperando a digitação de uma tecla
21.             Console.ReadKey();
22.
23.         }
24.     }
25. }
```

Execução



ReadLine – como receber um número real

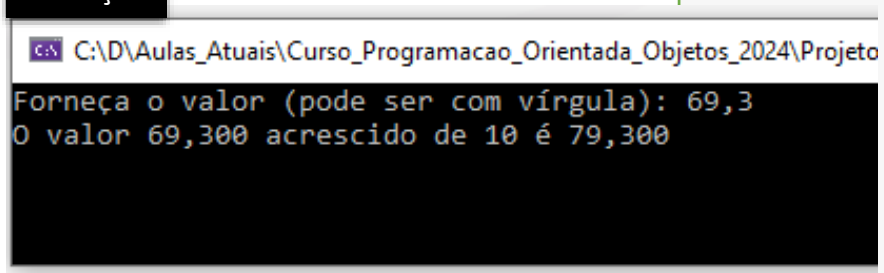
- Objetivo
  - O comando ReadLine, recebe variáveis do tipo string
  - Caso a variável à qual se deseja atribuir um comando digitado seja um número real (float, double, decimal). é necessário converter com o commando Parse
- Sintaxe

```
foidigitado = double.Parse(Console.ReadLine());
foidigitado = float.Parse(Console.ReadLine());
foidigitado = decimal.Parse(Console.ReadLine());
```

ReadLine – como receber um número real do tipo double

```
1.  namespace Aula05_Ex06
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // Declara Variáveis double
8.              double nd, resDouble;
9.
10.             // Recebe um valor
11.             Console.Write("Forneça o valor (pode ser com vírgula): ");
12.             nd = double.Parse(Console.ReadLine());
13.
14.             // Atribui valor
15.             resDouble = nd + 10;
16.
17.             // Informa os valores
18.             Console.WriteLine("O valor {0:F3} acrescido de 10 é {1:F3}", nd, resDouble);
19.
20.             // Mantem a tela aberta esperando a digitação de uma tecla
21.             Console.ReadKey();
22.         }
23.     }
24. }
```

Execução



ReadLine – como receber um número real do tipo float

Conforme pode ser observado, na linha “resFloat = nf + 10;”, não foi colocado a letra F após o 10 conforme no exemplo da aula anterior e a operação debug, não apresentou erro. Isto aconteceu porque o valor 10 é inteiro. Caso se desejasse utilizar um valor fracionário o sufixo F deveria ser acrescido.

```
1.  namespace Aula05_Ex07
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] arg
6.          {
7.              // Declara Variáveis floa
8.              float nf, resFloat;
9.
10.             // Recebe um valor
11.             Console.Write("Forneça o preço: ");
12.             nf = float.Parse(Console.ReadLine());
13.
14.             // Atribui valor
15.             resFloat = nf + 10;
16.
17.             // Informa os valores
18.             Console.WriteLine("O preço {0:C} acrescido de 10 é {1:C}", nf,
    resFloat);
19.
20.             // Mantem a tela aberta esperando a digitação de uma tecla
21.             Console.ReadKey();
22.         }
23.     }
24. }
```

Execução

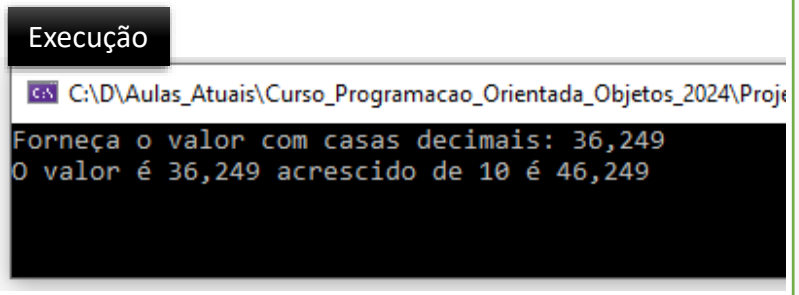
```
C:\D\Aulas_Atuais\Curso_Programacao_Orientada_Objetos_2024\Projetos
Forneça o preço: 36,90
O preço R$ 36,90 acrescido de 10 é R$ 46,90
```

ReadLine – como receber um número real do tipo decimal

Conforme pode ser observado, na linha “resDecimal = nd + 10;”, não foi colocado a letra M após o 10 conforme no exemplo da aula anterior e a operação debug, não apresentou erro.
É importante ficar atento, pois assim como no exemplo anterior (utilizando variável tipo float), o erro não ocorreu porque o valor 10 é inteiro, caso se utilizasse valores fracionários o sufixo M, (assim como o F do exemplo anterior) torna-se obrigatório.

```
1.  namespace Aula05_Ex08
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // Declara Variáveis decimal
8.              decimal nd, resDecimal;
9.
10.             // Recebe um valor
11.             Console.Write("Forneça o valor com casas decimais: ");
12.             nd = decimal.Parse(Console.ReadLine());
13.
14.             // Atribui valor
15.             resDecimal = nd + 10;
16.
17.             // Informa os valores
18.             Console.WriteLine("O valor é {0} acrescido de 10 é {1}", nd,
    resDecimal);
19.
20.             // Mantem a tela aberta esperando a digitação de uma tecla
21.             Console.ReadKey();
22.         }
23.     }
24. }
```

Execução



Conversão de conteúdos

- Objetivo
  - Algumas vezes pode ser necessário converter conteúdos dentro de um programa para que seja possível operar com estes valores. Isto pode ser feito com uma sintaxe de atribuição mesclada com conversão que também pode simultaneamente estar declarando a variável. O programa a seguir fornece exemplos destas declarações com atribuição e conversão simultânea

Convers

```
1.  namespace Aula05_Ex09
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // declara e inicializa variáveis do tipo string
8.              string valor_string1 = "530";
9.              string valor_string2 = "2345,32";
10.             string valor_string3 = "41,25";
11.             // declara variável para pegar o resultado
12.             double resultado;
13.
14.             // converte string para inteiro (declara também a variável int)
15.             int valor_integer = int.Parse(valor_string1);
16.             // exibe o resultado
17.             Console.WriteLine("Valor convertido para Inteiro: {0}", valor_integer);
18.
19.             // converte string para double (declara também a variável double)
20.             double valor_double = double.Parse(valor_string2);
21.             // exibe o resultado
22.             Console.WriteLine("Valor declarado como double: {0}", valor_double);
23.
24.             // converte string para float (declara também a variável float)
25.             float valor_float = float.Parse(valor_string3);
26.             // exibe o resultado (o \n antes de fechar as aspas acrescenta uma linha em branco)
27.             Console.WriteLine("Valor convertido para float: {0} \n", valor_float);
28.
29.             // Calcula e fornece o resultado
30.             resultado = valor_integer + valor_double + valor_float;
31.             Console.WriteLine("A soma dos valores calculados após convertidos é {0} ", resultado);
32.
33.             //Mantem a tela aberta esperando a digitação de uma tecla
34.             Console.ReadKey();
35.         }
36.     }
37. }
```

Execução

```
C:\D\Aulas_Atuais\Curso_Programacao_Orientada_Objetos_2024\Projetos_POO\Aula0
Valor convertido para Inteiro: 530
Valor declarado como double: 2345,32
Valor convertido para float: 41,25

A soma dos valores calculados após convertidos é 2916,57
```

Declarações com atribuições simultâneas.

Divisão de números declarados como inteiros

- Objetivo
  - No C#, cada tipo de dados possui seus próprios operadores, isto significa que, quando se divide dois números inteiros, o resultado será sempre a parte inteira da divisão.

  - Caso se deseje que o resultado seja um número Real (do tipo float ou double), é melhor que os valores que serão divididos, sejam também números reais, entretanto.
  - Caso por qualquer razão os valores que serão divididos precisarem ser declarados como inteiros e os resultados das divisões precisarem ser números reais, é possível no momento da divisão informar o compilador, que o resultado da divisão deve ser tratado como real no momento da divisão, isto é feito com a sintaxe a seguir

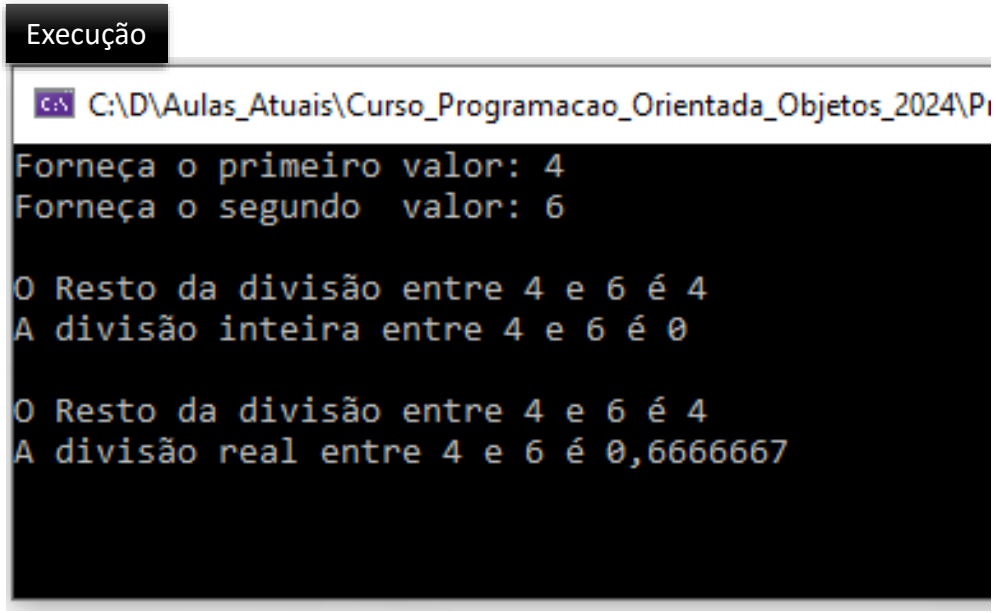
- Sintaxes

```
double(resultado) = double(dividendo)/double(divisor);
float(resultado) = float(dividendo)/float(divisor);
```

Divisão

```
1.  namespace Aula05_Ex10
2.  {
3.      internal class Program
4.      {
5.          static void Main(string[] args)
6.          {
7.              // Declara variaveis
8.              int v1, v2, restoInt, divInt;
9.              float restoReal, divReal;
10.
11.             // Solicita valores
12.             Console.Write("Forneça o primeiro valor: ");
13.             v1 = int.Parse(Console.ReadLine());
14.
15.             Console.Write("Forneça o segundo  valor: ");
16.             v2 = int.Parse(Console.ReadLine());
17.
18.             restoInt = v1 % v2;
19.             divInt = v1 / v2;
20.             restoReal = (float)v1 % (float)v2;
21.             divReal = (float)v1 / (float)v2;
22.             // Fornece os valores
23.             Console.WriteLine();
24.             Console.WriteLine("O Resto da divisão entre {0} e {1} é {2} ", v1, v2, restoInt);
25.             Console.WriteLine("A divisão inteira entre {0} e {1} é {2} ", v1, v2, divInt);
26.             Console.WriteLine();
27.             Console.WriteLine("O Resto da divisão entre {0} e {1} é {2} ", v1, v2, restoReal);
28.             Console.WriteLine("A divisão real entre {0} e {1} é {2} ", v1, v2, divReal);
29.
30.             // Mantem a tela aberta esperando a digitação de uma tecla
31.             Console.ReadKey();
32.         }
33.     }
34. }
```

Execução



EAD
UNISANTA