# VIANNSTITUTO JUNIOR



# Algoritmos

Sub-rotinas

Professor: Camillo Falcão



# Estruturação de programas

 O desenvolvimento de um programa é um processo dinâmico. Frequentemente, é necessário rever códigos para corrigir erros, fazer atualizações ou utilizar como base para resolução de outro problema.

 Por isso, é fundamental que um programa seja escrito de forma organizada, facilitando sua manutenção, reuso e/ou adaptação.



## Sub-rotinas

- Uma sub-rotina é um trecho de código computacional que realiza uma tarefa bem definida. Exemplos:
  - Calcular área de um círculo.
  - · Converter graus Celsius para Fahrenheit.
- Uma única sub-rotina pode ser executada várias vezes em um mesmo programa. Pode também ser executada por diferentes programas.



## Sub-rotinas

- Você já utiliza sub-rotinas em seus programas:
  - WriteLine é uma sub-rotina da classe Console que realiza a tarefa de imprimir na tela do computador
  - ReadLine é uma sub-rotina que lê valores do teclado e retorna o valor lido para que o mesmo possa ser utilizado
  - Sqrt é uma sub-rotina da classe Math que calcula e devolve o valor da raiz quadrada de um número
  - Pow é uma sub-rotina da classe Math que calcula e retorna o valor da potência, dados base e expoente indicados



# Sub-rotinas

- Repare que você não precisa saber como as subrotinas WriteLine, ReadLine, Sqrt e Pow foram implementadas, precisa saber apenas:
  - o que a sub-rotina faz, isto é, qual a tarefa executada por ela
  - como ela pode ser utilizada (sua sintaxe)



# Vantagem do uso de subrotinas

- Uma mesma tarefa pode ser implementada uma única vez e ser utilizada várias vezes (por um ou mais programas):
  - erros precisam ser corrigidos em um único lugar
  - a reutilização da sub-rotina (chamada) pode ser feita de forma simples
  - o código fica legível, mais fácil de ser entendido e mais compacto
  - o uso de sub-rotinas possibilita a modularização do código de um programa, isto é, o desenvolvimento do código organizado em módulos funcionais



# Programando sub-rotinas

- Existem três situações distintas no processo de programação envolvendo sub-rotinas:
  - 1) Declaração: a declaração de uma subrotina especifica sua sintaxe.
    Permite que o usuário que não implementou
    a sub-rotina e quer utilizá-la conheça seu
    nome, os valores que necessita e o valor que
    será retornado.



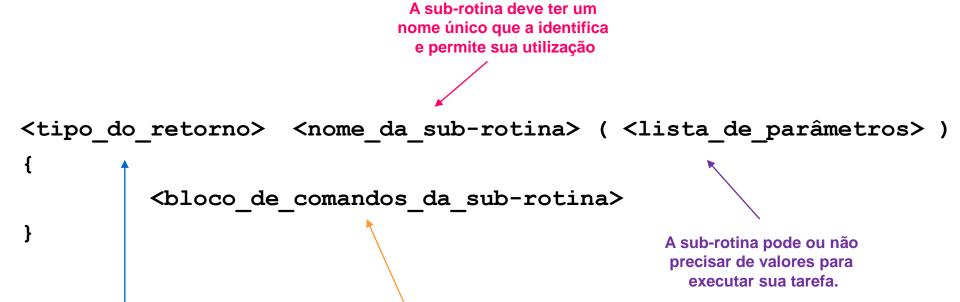
# Programando sub-rotinas

- 2) Definição: a definição de uma sub-rotina é a implementação da mesma. É uma extensão da declaração, dado que, além da sintaxe, contém o código que realiza a sua tarefa.
- 3) Chamada: a chamada à uma sub-rotina é a utilização da mesma dentro de alguma outra sub-rotina ou programa. Uma sub-rotina não funciona sozinha, precisa ser chamada cada vez que sua execução for necessária.



A sub-rotina pode ou não produzir algum valor como resultado.

# Definição de uma subrotina



Bloco de comandos que implementam a funcionalidade executada pela sub-rotina



```
static double CelsiusFahrenheit ( double tc )
 double tf;
 tf = 1.8 * tc + 32;
  return tf;
static void Main(string[] args)
 double cels;
 double fahr;
  Console.Write("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit (cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
```



```
static double CelsiusFahrenheit ( double tc )
                             tipoDeRetorno nomeDasub-rotina (parametros)
 double tf;
                             {
  tf = 1.8 * tc + 32;
                                bloco de comandos da sub-rotina
  return tf;
static void Main(string[] args)
 double cels;
 double fahr:
  Console.Write ("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
```



```
static double CelsiusFahrenheit ( doub resultado.
 double tf;
  tf = 1.8 * tc + 32;
  return tf;
static void Main(string[] args)
                                        ser void.
 double cels;
 double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
```

A sub-rotina pode ou não produzir um valor como

Quando um valor é produzido, seu tipo deve ser indicado.

Se não há retorno de valor, o tipo de retorno informado deve



```
static double CelsiusFahrenheit ( doub
{
  double tf;
  tf = 1.8 * tc + 32;
  return tf;
}
```

#### O nome da sub-rotina:

- precisa ser único
- deve seguir as mesmas regras de identificadores de variáveis
- deve identificar a ação executada pela sub-rotina

```
static void Main(string[] args)
{
   double cels;
   double fahr;
   Console.Write("Entre com temperatura em Celsius: ");
   cels = Convert.ToDouble(Console.ReadLine());
   fahr = CelsiusFahrenheit(cels);
   Console.Write("Temperatura em Fahrenheit:");
   Console.Write(fahr);
}
```



```
static double CelsiusFahrenheit ( double tc )
 double tf;
 tf = 1.8 * tc + 32;
  return tf;
static void Main(string[] args)
 double cels;
 double fahr;
  Console.Write ("Entre com temperatura em Celsius no bloco da sub-
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
```

A lista de parâmetros deve indicar os valores que precisam ser fornecidos quando a sub-rotina for chamada. Para cada valor (parâmetro), devem ser especificados:

- tipo do parâmetro
- nome do parâmetro

rotina



```
static double CelsiusFahrenheit ( double
  double tf;
  tf = 1.8 * tc + 32; \leftarrow
  return tf;
static void Main(string[] args)
  double cels;
  double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
```

Console.Write ("Temperatura em Fahrenheit:");

Console.Write(fahr);

O bloco de comandos deve ter, em seu início, a declaração das variáveis necessárias no código, sem incluir os identificadores da lista de parâmetros.

Só depois devem aparecer os comandos que implementam a tarefa a ser executada pela sub-rotina.



O bloco de comandos deve

```
static double CelsiusFahrenheit ( double
                                        terminar com um comando
                                        de retorno, seguido do valor
 double tf;
                                         resultante da tarefa
 tf = 1.8 * tc + 32;
  return tf; ←
                                        executada pela sub-rotina.
                                        Se a sub-rotina não produzir
static void Main(string[] args)
                                         um valor, o comando de
 double cels;
                                         retorno pode ser omitido.
 double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
 cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
 Console.Write(fahr);
```



```
static double CelsiusFahrenheit ( double tc )
 double tf;
                                         Toda sub-rotina precisa ser
 tf = 1.8 * tc + 32;
  return tf;
                                         definida ou declarada antes
                                         de ser chamada.
static void Main(string[] args)
 double cels;
 double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
```



```
using System;
static void Main(string[] args)
  double cels;
  double fahr;
  Console. Write ("Entre com temperatura e
  cels = Convert. To Double (Console. ReadLi
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenhe
  Console.Write(fahr);
static double CelsiusFahrenheit ( dc
  double tf;
  tf = 1.8 * tc + 32;
  return tf;
```

As sub-rotinas
Console.Write,
Console.WriteLine e
Console.ReadLine podem
ser utilizadas porque a
biblioteca que contém sua
declaração foi referenciada
no início do arquivo, antes
do programa.



# Definição de uma sub-

```
static void Main(string[] args)
 double cels;
 double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
 cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit(cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
static double CelsiusFahrenheit
  double tf;
  tf = 1.8 * tc + 32;
  return tf;
```

A chamada de uma subrotina deve incluir somente o seu *nome* e o *valor* de dc cada parâmetro.

O tipo de retorno e o tipo dos parâmetros <u>não</u> aparecem na chamada.



```
static void Mai nome_da_sub-rotina ( lista_com_um_valor_para_cada_parametro );
  double cels;
  double fahr;
  Console.Write ("Entre com temperatura em Celsius: ");
  cels = Convert.ToDouble(Console.ReadLine());
  fahr = CelsiusFahrenheit (cels);
  Console.Write ("Temperatura em Fahrenheit:");
  Console.Write(fahr);
static double CelsiusFahrenheit ( double tc )
  double tf;
  tf = 1.8 * tc + 32;
  return tf;
```



# Definição de uma sub-

```
static void Main(string[] args)
 double cels;
 double fahr;
  Console.Write ("Entre com temperatura e
  cels = Convert. To Double (Onsole. ReadLi
  fahr = CelsiusFahrenheit (cels);
  Console.Write ("Temperatura em Fahrenhe
  Console.Write(fahr);
static double CelsiusFahrenheit
  double tf;
  tf = 1.8 * tc + 32;
  return tf;
```

No caso da sub-rotina
CelsiusFahrenheit, o
único parâmetro, tc, tem
tipo double. Então, um
valor do tipo double precisa
ser passado como
parâmetro.

Na chamada, o valor armazenado pela variável cels é passado como parâmetro para a sub-rotina.



1 2		linha	cels	fahr	tc	tf
3						
4						
5 <b>s</b>	tatic void Main(string[] args) {					
6	double cels;					
7	double fahr;					
8						
9	Console.Write("Temperatura em Celsius: ");					
10	<pre>cels = Convert.ToDouble(Console.ReadLine());</pre>					
11						
12	<pre>fahr = CelsiusFahrenheit(cels);</pre>					
13						
14	Console.Write("Temperatura em Fahrenheit:");					
15	Console.Write(fahr);					
16						
17						
18	}					
19						
	<pre>static double CelsiusFahrenheit(double tc) {</pre>					
21	double tf;					
	tf = 1.8 * tc + 32;					
23	return tf;					
24	}					



24 }

# Execução de uma subrotina

#### static void Main(string[] args) double cels; 6 double fahr; Console.Write("Temperatura em Celsius: "); 10 cels = Convert.ToDouble(Console.ReadLine()); 11 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; tf = 1.8 \* tc + 32;22 23 return tf;

linha	cels	fahr	tc	tf
5	?	?		



2223

24 }

double tf;

return tf;

tf = 1.8 \* tc + 32;

# Execução de uma subrotina

#### static void Main(string[] args) { double cels; double fahr; 9 Console.Write("Temperatura Celsius: "); 10 cels = Convert.ToDouble(Console.ReadLine()); 11 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) {

#### **TESTE DE MESA**

	TESTE DE MESA						
linha	cels	fahr	tc	tf			
5	?	?					



#### **TESTE DE MESA**

tf

tc

```
linha
                                                           cels
                                                                  fahr
3
                                                      5
                                                           20.0
                                                     10
  static void Main(string[] args) {
    double cels;
    double fahr:
    Console.Write("Temperatura em Celsius: ");
10
    cels = Convert.ToDouble(Console.ReadLine());
11
12
    fahr = CelsiusFahrenheit(cels);
13
    Console.Write("Temperatura em Fahrenheit:");
14
15
    Console.Write(fahr);
16
17
18
19
20 static double CelsiusFahrenheit (double tc)
2.1
     double tf;
                                  Entre com temperatura em Celsius: 20
22
  tf = 1.8 * tc + 32;
2.3
     return tf;
24 }
```



```
cels
                                                                  fahr
                                                     linha
                                                                                tf
                                                                          tc
                                                      5
  static void Main(string[] args) {
                                                           20.0
                                                      10
    double cels;
6
                                                      12
                                                           20.0
    double fahr:
    Console.Write("Temperatura em Celsius: ");
10
    cels = Convert.ToDouble(Console.ReadLine());
11
    fahr = CelsiusFahrenheit(cels);
12
13
14
    Console.Write("Temperatura em Fahrenheit:");
15
    Console.Write(fahr);
16
17
18
19
   static double CelsiusFahrenheit(double tc) {
     double tf;
2.1
                                  Entre com temperatura em Celsius: 20
22
   tf = 1.8 * tc + 32;
2.3
     return tf;
24 }
```



```
1
4
 static void Main(string[] args) {
    double cels;
    double fahr;
    Console.Write("Temperatura em Celsius: ");
10
    cels = Convert.ToDouble(Console.ReadLine());
11
12
    fahr = CelsiusFahrenheit(cels);
13
14
    Console.Write ("Temperatura em Fahrenheit:");
15
   Console.Write(fahr);
16
17
18 }
19
20 static double CelsiusFahrenheit (double tc)
```

```
linha
       cels
              fahr
                      tc
                              tf
 5
      20.0
 10
               ?
 12
      20.0
                     20.0
20
```

```
21 double tf;

22 tf = 1.8 * tc + 32;

23 return tf;

24 }
```

```
Entre com temperatura em Celsius: 20
```



22

2.3

24 }

# Execução de uma subrotina

```
static void Main(string[] args) {
    double cels;
6
    double fahr:
    Console.Write("Temperatura em Celsius: ");
    cels = Convert.ToDouble(Console.ReadLine());
10
11
    fahr = CelsiusFahrenheit(cels);
13
    Console.Write("Temperatura em Fahrenheit:");
14
15
    Console.Write(fahr);
16
17
18
19
   static double CelsiusFahrenheit(double tc) {
21
     double tf;
```

tf = 1.8 \* tc + 32;

return tf;

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0

```
Entre com temperatura em Celsius: 20
```



#### static void Main(string[] args) { double cels; 6 double fahr: Console.Write("Temperatura em Celsius: "); 10 cels = Convert.ToDouble(Console.ReadLine()); 11 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; 22 tf = 1.8 \* tc + 32;23 return tf; 24 }

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0



24 }

# Execução de uma subrotina

#### static void Main(string[] args) { double cels; double fahr: Console.Write("Temperatura em Celsius: "); cels = Convert.ToDouble(Console.ReadLine()); 10 11 fahr = CelsiusFahrenheit(cels); 12 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; tf = 1.8 \* tc + 32;22 2.3 return tf;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	?		



24 }

return tf;

# Execução de uma subrotina

#### static void Main(string[] args) { double cels; double fahr; Console.Write("Temperatura em Celsius: "); 10 cels = Convert.ToDouble(Console.ReadLine()); 11 12 fahr = 68;13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; tf = 1.8 \* tc + 32;22

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	?		



24 }

# Execução de uma subrotina

#### static void Main(string[] args) { double cels; double fahr; Console.Write("Temperatura em Celsius: "); cels = Convert.ToDouble(Console.ReadLine()); 10 11 fahr = CelsiusFahrenheit(cels); 12 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; tf = 1.8 \* tc + 32;22 2.3 return tf;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	68.0		



24 }

return tf;

# Execução de uma subrotina

#### static void Main(string[] args) { double cels; double fahr: Console.Write("Temperatura em Celsius: "); cels = Convert.ToDouble(Console.ReadLine()); 10 11 fahr = CelsiusFahrenheit(cels); 13 14 Console.Write("Temperatura em Fahrenheit:") 15 Console.Write(fahr); 16 17 18 19 static double CelsiusFahrenheit(double tc) { 2.1 double tf; 22 tf = 1.8 \* tc + 32;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	68.0		

Entre com temperatura em Celsius: 20 Temperatura em Fahrenheit:



24 }

return tf;

# Execução de uma subrotina

#### 1 static void Main(string[] args) { double cels; 6 double fahr; 8 Console.Write("Temperatura em Celsius: "); 10 cels = Convert.ToDouble(Console.ReadLine()); 11 12 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 20 **static double** CelsiusFahrenheit (**double** tc) { 2.1 double tf; 22 tf = 1.8 \* tc + 32;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	68.0		

Entre com temperatura em Celsius: 20 Temperatura em Fahrenheit: 68.000000



24 }

return tf;

# Execução de uma subrotina

#### 1 static void Main(string[] args) { double cels; 6 double fahr; 8 Console.Write("Temperatura em Celsius: "); cels = Convert.ToDouble(Console.ReadLine()); 10 11 12 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 20 **static double** CelsiusFahrenheit (**double** tc) { 2.1 double tf; 22 tf = 1.8 \* tc + 32;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf
5	?	?		
10	20.0	?		
12	20.0	?		
20			20.0	?
22			20.0	68.0
12	20.0	68.0		

Entre com temperatura em Celsius: 20 Temperatura em Fahrenheit: 68.000000



2.3

24 }

return tf;

### Execução de uma subrotina

#### 1 static void Main(string[] args) { double cels; 6 double fahr; 8 Console.Write("Temperatura em Celsius: "); cels = Convert.ToDouble(Console.ReadLine()); 10 11 12 fahr = CelsiusFahrenheit(cels); 13 Console.Write("Temperatura em Fahrenheit:"); 14 15 Console.Write(fahr); 16 17 18 19 20 **static double** CelsiusFahrenheit (**double** tc) { 2.1 double tf; 22 tf = 1.8 \* tc + 32;

#### **TESTE DE MESA**

linha	cels	fahr	tc	tf	
_		0			
5	?	?			
10	20.0	?			
12	20.0	?			
20			20.0	?	
22			20.0	68.0	
12	20.0	68.0			

Entre com temperatura em Celsius: 20 Temperatura em Fahrenheit: 68.000000



### Tipos de sub-rotina

 Procedimento: sub-rotina que não retorna valor para o programa ou sub-rotina que a chamou.

 Um procedimento normalmente executa uma ação que não gera dados.



```
static void ImprimeFormatoHora (int gtdMinutos)
2
3
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
    Console. Write ("{0}:{1}", hora, min);
 static void Main(string[] args)
10 {
11
     int minutos;
12
    Console.Write("Minutos para converter em hora:");
13
     minutos = Convert. ToInt32 (Console. ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console. ReadKey();
17 }
```



```
static void ImprimeFormatoHora(int qtdMinutos)
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
                                                       <u>Tipo de retorno</u>
    Console. Write ("{0}:{1}", hora, min);
                                                       Como não há retorno
                                                       de valor algum, o tipo
  static void Main(string[] args)
                                                       indicado é void.
10 {
     int minutos;
11
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
    ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```



```
1 static void ImprimeFormatoHora(int gtdMinutos)
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
    Console. Write ("\{0\}:\{1\}", hora, min);
    return;
   static void Main(string[] args)
11
12
     int minutos;
13
     Console.Write ("Minutos para converter em hora:");
     minutos = Convert.ToInt32(Console.ReadLine());
14
1.5
     Console.Write("{0} minutos equivale a ", minutos);
16
     ImprimeFormatoHora(minutos);
17
     Console. ReadKey();
18 }
```

### Comando de retorno

Como não há retorno de valor algum, o comando return aparece sozinho (apenas seguido do ponto e vírgula) ou é omitido.



```
1 static void ImprimeFormatoHora(int gtdMinutos)
2 {
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
    Console.Write("{0}:{1}", hora, min);
   return;
    Console.Write("Esta linha nunca sera executada! ");
10
   static void Main(string[] args)
12 {
     int minutos;
13
14
     Console.Write("Minutos para converter em hora:");
     minutos = Convert.ToInt32(Console.ReadLine());
15
16
     Console.Write("{0} minutos equivale a ", minutos);
17
     ImprimeFormatoHora(minutos);
18
     Console.ReadKey();
19 }
```

### Comando de retorno

Cuidado ao usar o comando return no meio de uma subrotina, pois, mesmo que haja comandos depois, a sub-rotina é encerrada assim que o comando return for executado.



```
static void ImprimeFormatoHora(int gtdMinutos)
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
    Console. Write ("{0}:{1}", hora, min);
  static void Main(string[] args)
10 {
11
     int minutos;
     Console.Write ("Minutos para converter em hora:");
12
13
     minutos = Convert. ToInt32 (Consolæ. ReadLine());
     Console.Write("{0} minutos equivale a ", minutos);
14
15
    ImprimeFormatoHora(minutos)
16
     Console.ReadKey();
17 }
```

#### Chamada

A chamada a um procedimento normalmente aparece sozinha em uma linha de comando.



1 static void ImprimeFormatoHora(int qtdMinutos) 2 {	linha	Minutos	hora	min	qtdMinuto s
3 int hora, min;					
4 hora = qtdMinutos / 60;					
5 min = qtdMinutos % 60;					
6 Console.Write("{0}:{1}", hora, min);					
7 }					
8					
9 static void Main(string[] args)					
10 {					
11 int minutos;					
12 Console.Write("Minutos para converter em hora:");					
<pre>13 minutos = Convert.ToInt32(Console.ReadLine());</pre>					
14 Console.Write("{0} minutos equivale a ", minutos);					
15 ImprimeFormatoHora(minutos);					
16 Console.ReadKey();					
17 }					



```
qtdMinuto
1 static void ImprimeFormatoHora (int qtdMinutos)
                                                          linha minutos hora min
    int hora, min;
   hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
    Console. Write ("\{0\}:\{1\}", hora, min);
9 static void Main(string[] args)
10 {
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
     Console.Write("{0} minutos equivale a ", minutos);
14
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```



#### **TESTE DE MESA**

```
atdMinuto
1 static void ImprimeFormatoHora(int gtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
  hora = qtdMinutos / 60;
   min = qtdMinutos % 60;
    Console.Write("{0}:{1}", hora, min);
7 }
8
9 static void Main(string[] args)
10 {
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console. ReadKey();
17 }
```

Entre com os minutos:



#### **TESTE DE MESA**

```
atdMinuto
1 static void ImprimeFormatoHora(int qtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
  hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
                                                           13
                                                                345
    Console. Write ("{0}:{1}", hora, min);
7 }
8
9 static void Main(string[] args)
10 {
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert. ToInt32 (Console. ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console. ReadKey();
17 }
```

Entre com os minutos: 345



```
atdMinuto
1 static void ImprimeFormatoHora(int qtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
  hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
                                                          13
                                                               345
    Console. Write ("{0}:{1}", hora, min);
7 }
8
9 static void Main(string[] args)
10 {
11
     int minutos;
     Console.Write ("Minutos para converter em hora:");
12
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console. ReadKey();
17 }
```

```
Entre com os minutos: 345
345 minutos equivale a
```



```
atdMinuto
1 static void ImprimeFormatoHora(int qtdMinutos)
                                                         linha minutos hora min
2
    int hora, min;
  hora = qtdMinutos / 60;
   min = qtdMinutos % 60;
                                                          13
                                                               345
    Console.Write("{0}:{1}", hora, min);
7 }
8
9 static void Main(string[] args)
10 {
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

```
Entre com os minutos: 345
345 minutos equivale a
```



#### **TESTE DE MESA**

```
atdMinuto
1 static void ImprimeFormatoHora(int qtdMinutos)
                                                         linha minutos hora min
2
    int hora, min;
  hora = qtdMinutos / 60;
   min = qtdMinutos % 60;
                                                          13
                                                               345
    Console. Write ("{0}:{1}", hora, min);
                                                                                345
7 }
8
9 static void Main(string[] args)
10 {
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
    minutos = Convert.ToInt32(Console.ReadLine());
14
    Console.Write("{0} minutos equivale a ", minutos);
15
    ImprimeFormatoHora(minutos);
16
     Console. ReadKey();
17 }
```

Entre com os minutos: 345
345 minutos equivale a



#### **TESTE DE MESA**

```
atdMinuto
1 static void ImprimeFormatoHora(int gtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
    hora = qtdMinutos / 60;
                                                           13
                                                               345
    min = qtdMinutos % 60;
    Console.Write("{0}:{1}", hora, min);
                                                                                345
7 }
8
                                                           4
                                                                                345
9 static void Main(string[] args)
10 {
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

Entre com os minutos: 345 345 minutos equivale a



#### **TESTE DE MESA**

```
atdMinuto
1 static void ImprimeFormatoHora(int gtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
    hora = qtdMinutos / 60;
                                                           13
                                                               345
    min = qtdMinutos % 60;
    Console.Write("{0}:{1}", hora, min);
                                                                                345
7
8
                                                           4
                                                                                345
9 static void Main(string[] args)
10 {
                                                           5
                                                                          45
                                                                                345
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

Entre com os minutos: 345 345 minutos equivale a



```
atdMinuto
1 static void ImprimeFormatoHora(int gtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
    hora = qtdMinutos / 60;
                                                           13
                                                               345
    min = qtdMinutos % 60;
    Console.Write("{0}:{1}", hora, min);
                                                                                345
7 }
8
                                                                                345
                                                           4
9 static void Main(string[] args)
10 {
                                                           5
                                                                          45
                                                                                345
11
     int minutos;
     Console.Write ("Minutos para converter em hora:");
12
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

```
Entre com os minutos: 345 345 minutos equivale a 5:45
```



```
atdMinuto
1 static void ImprimeFormatoHora(int qtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
   hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
                                                           13
                                                               345
    Console. Write ("{0}:{1}", hora, min);
                                                                                345
7 }
8
                                                                                345
                                                           4
9 static void Main(string[] args)
10 {
                                                                                345
                                                           5
                                                                          45
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
     Console.Write("{0} minutos equivale a ", minutos);
14
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

```
Entre com os minutos: 345
345 minutos equivale a 5:45
```



```
atdMinuto
1 static void ImprimeFormatoHora(int gtdMinutos)
                                                          linha minutos hora min
2
    int hora, min;
    hora = qtdMinutos / 60;
    min = qtdMinutos % 60;
                                                           13
                                                                345
    Console. Write ("{0}:{1}", hora, min);
                                                                                 345
7 }
8
                                                                                 345
                                                           4
9 static void Main(string[] args)
10 {
                                                            5
                                                                           45
                                                                                 345
11
     int minutos;
12
     Console.Write ("Minutos para converter em hora:");
13
     minutos = Convert.ToInt32(Console.ReadLine());
14
     Console.Write("{0} minutos equivale a ", minutos);
15
     ImprimeFormatoHora(minutos);
16
     Console.ReadKey();
17 }
```

```
Entre com os minutos: 345 345 minutos equivale a 5:45
```



### Tipos de sub-rotina

 Função: sub-rotina que retorna um único valor para o programa ou sub-rotina que a chamou.

 Funções são utilizadas para realizar uma operação e retornam alguma resposta relativa à operação realizada.



```
3 static int ConverteEmMinutos(int hora, int min)
4
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos;
9
   static void Main(string[] args)
11
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."},
18
            hora, min, qtdMinutos);
19
```



```
static int ConverteEmMinutos(int hora, int min)
4
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos;
9
   static void Main(string[] args)
11
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert.ToInt32(Console.ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     qtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos.}",
18
             hora, min, qtdMinutos);
19 }
```

#### Tipo de retorno

O tipo informado deve ser definido com base no dado produzido pela função. Neste caso, o valor que a função vai retornar deve ser um número inteiro.



```
static int ConverteEmMinutos(int hora, int min)
4
    int totalMinutos;
    totalMinutos = hora * 60 + min
    return totalMinutos;
9
   static void Main(string[] args)
11
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     qtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."},
18
             hora, min, qtdMinutos);
19 }
```

#### Comando de retorno

O valor que será informado no retorno da função precisa ser indicado junto ao comando return.

Neste caso, este valor corresponde ao conteúdo da variável total Minutos.



```
static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos;
9
   static void Main(string[] args)
11
12
     int hora, min, qtdMinutos;
13
     Console.Write ("Horas e minatos:");
     hora = Convert.ToInt32(Console.ReadLine());
14
15
     min = Convert.ToInt32 Console.ReadLine());
16
     qtdMinutos = ConverteEmMinutos(hora, min);
     Console. Write ("\{0\}:\{1\} = \{2\}  minutos.",
17
18
            hora, min, qtdMinutos);
19
```

#### **Chamada**

Como a função, ao ser executada, retorna um valor para a sub-rotina que a chamou, este valor normalmente é armazenado em uma variável de mesmo tipo (recomendável para quem está começando a programar).



```
static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos;
9
   static void Main(string[] args)
11
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minytos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32 (Console.ReadLine());
16
     qtdMinutos = ConverteEmMinutos(hora, min);
     Console. Write ("\{0\}:\{1\} = \{2\}  minutos.",
17
18
            hora, min, qtdMinutos);
19 }
```

#### Chamada

Outras opções são imprimir o valor de retorno, utilizar o valor em uma expressão, ou utilizá-lo em qualquer outra situação onde um valor do mesmo tipo é esperado.



```
linha hora min hora min total Minutos
3 static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
   static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
     hora = Convert.ToInt32(Console.ReadLine());
14
1.5
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."})
18
            hora, min, qtdMinutos);
19 }
```



#### **TESTE DE MESA** linha hora min hora min total Minutos 3 static int ConverteEmMinutos(int hora, int min) 10 int totalMinutos; totalMinutos = hora \* 60 + min; return totalMinutos: 10 **static void** Main(**string**[] args) 11 { 12 int hora, min, qtdMinutos; 13 Console.Write("Horas e minutos:"); hora = Convert.ToInt32(Console.ReadLine()); 14 15 min = Convert.ToInt32(Console.ReadLine()); 16 gtdMinutos = ConverteEmMinutos(hora, min); 17 Console.Write( $((0):\{1\}) = \{2\}$ minutos. 18 hora, min, qtdMinutos); 19 }



```
linha hora min hora min total Minutos
3 static int ConverteEmMinutos(int hora, int min)
                                                     10
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
  static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console.Write("\{0\}:\{1\} = \{2\} minutos.",
18
            hora, min, qtdMinutos);
19 }
Entre com horas e minutos:
```



10

14 5

#### **TESTE DE MESA**

linha hora min hora min totalMinutos

```
3 static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
   static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
     qtdMinutos = ConverteEmMinutos(hora, min);
16
     Console.Write("\{0\}:\{1\} = \{2\} minutos.",
17
18
            hora, min, qtdMinutos);
19 }
```

Entre com horas e minutos: 5



#### 3 static int ConverteEmMinutos (int hora, int mir int totalMinutos; totalMinutos = hora \* 60 + min; return totalMinutos:

```
11 {
12
     int hora, min, qtdMinutos;
```

static void Main(string[] args)

Console.Write ("Horas e minutos:"); 13

14 hora = Convert.ToInt32 (Console.ReadLine())

15 min = Convert.ToInt32(Console.ReadLine()); 16 gtdMinutos = ConverteEmMinutos(hora, min);

Console. Write (" $\{0\}$ :  $\{1\}$  =  $\{2\}$  minutos.",

17 18 hora, min, qtdMinutos);

19 }

```
Entre com horas e minutos: 5
45
```

#### TECTE DE MECA

TESTE DE MESA						
	linha	hora	min	hora	min	totalMinutos
n)	10	?	?			
	14	5	45			
<u>;</u>						
╛						



#### **TESTE DE MESA**

5 45

```
linha hora min hora min total Minutos
3 static int ConverteEmMinutos(int hora, int min)
                                                     10
    int totalMinutos;
                                                     14
    totalMinutos = hora * 60 + min;
    return totalMinutos:
10 static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert.ToInt32(Console.ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}: \{1\} = \{2\} minutos.",
18
            hora, min, qtdMinutos);
19 }
```

```
Entre com horas e minutos: 5
45
```



# TESTE DE MESA linha hora min hora min totalMinutos

```
3 static int ConverteEmMinutos (int hora, int min)
4 {
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
   static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
1 4
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."})
18
            hora, min, qtdMinutos);
19 }
```

```
10
14
   5 45
3
          5 45
```

Entre com horas e minutos: 5
45



```
3 static int ConverteEmMinutos(int hora, int min)
  int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos;
10 static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."})
18
            hora, min, qtdMinutos);
19 }
```

Entre com horas e minutos: 5

45

```
TESTE DE MESA
linha hora min hora min total Minutos
10
14
     5
        45
 3
             5 45
                45
                        345
```



45

## Execução de uma função

```
TESTE DE MESA
                                                    linha hora min hora min total Minutos
3 static int ConverteEmMinutos(int hora, int min)
                                                     10
    int totalMinutos;
                                                     14
                                                          5
                                                             45
    totalMinutos = hora * 60 + min;
                                                     3
                                                                  5 45
    return totalMinutos;
8 }
                                                                      45
                                                     6
                                                                             345
  static void Main(string[] args)
11 {
12
     int hora, min, gtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert.ToInt32(Console.ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos.}",
18
            hora, min, qtdMinutos);
19 }
Entre com horas e minutos: 5
```



```
3 static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
  static void Main(string[] args)
11 {
12
     int hora, min, qtdMinutos;
13
     Console.Write("Horas e minutos:");
14
     hora = Convert.ToInt32(Console.ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos.}",
18
            hora, min, qtdMinutos);
19 }
```

Entre com horas e minutos: 5

45

```
TESTE DE MESA
linha hora min hora min total Minutos
10
14
     5 45
 3
             5
                45
             5 45
                       345
16
       45
     5
```



```
3 static int ConverteEmMinutos (int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
  static void Main(string[] args)
11 {
12
     int hora, min, gtdMinutos;
13
     Console.Write("Horas e minutos:");
     hora = Convert. ToInt32 (Console. ReadLine());
14
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
     Console.Write("\{0\}:\{1\} = \{2\} minutos.",
17
18
            hora, min, qtdMinutos);
19 }
```

```
Entre com horas e minutos: 5
45
5:45 = 345 minutos.
```

		ILSIL DL WILSA						
	linha	hora	min	hora	min	totalMinutos		
)	10	?	?					
	14	5	45					
	3			5	45	?		
	6			5	45	345		
	15	5	45					
_								



# Execução de uma função

```
3 static int ConverteEmMinutos(int hora, int min)
    int totalMinutos;
    totalMinutos = hora * 60 + min;
    return totalMinutos:
  static void Main(string[] args)
11 {
12
     int hora, min, gtdMinutos;
13
     Console.Write("Horas e minutos:");
1 4
     hora = Convert. ToInt32 (Console. ReadLine());
15
     min = Convert.ToInt32(Console.ReadLine());
16
     gtdMinutos = ConverteEmMinutos(hora, min);
17
     Console. Write ("\{0\}:\{1\} = \{2\} \text{ minutos."})
18
            hora, min, qtdMinutos);
19 }
```

```
TESTE DE MESA
linha hora min hora min total Minutos
10
14
     5 45
 3
             5
                45
                45
                        345
15
        45
     5
```

```
Entre com horas e minutos: 5
45
5:45 = 345 minutos.
```



 Durante a criação de um programa, quando é identificada uma tarefa que pode ser codificada como uma sub-rotina, é necessário primeiramente analisar a sua SEMÂNTICA para que então seja possível definir a sua SINTAXE.



**SEMÂNTICA:** refletindo sobre a tarefa a ser executada pela sub-rotina, é necessário definir:

- Quais dados de entrada são necessários?
- A tarefa deve produzir um resultado como retorno?



**SINTAXE:** após definir os dados de entrada e o resultado da sub-rotina, é necessário especificar o tipo de cada um dos parâmetros e o tipo de retorno.

- Se a sub-rotina não tem parâmetro, ainda assim os parênteses são necessários (mesmo sem conteúdo algum).
- Se não há retorno de valor, o tipo é void.



- Depois de definir a declaração (ou protótipo) da sub-rotina, o procedimento utilizado para codificar seu bloco de comandos é idêntico ao que você já utilizava para criar programas:
  - identificar as variáveis adicionais necessárias;
  - declarar tais variáveis;
  - fazer refinamentos ou utilizar o método direto para implementar a tarefa necessária.
- Ao final, deve-se usar o comando de retorno.



- 1) Faça um procedimento que receba três valores reais e imprima a média aritmética desses valores.
- 2) Faça uma função que receba o valor de um produto e um percentual de desconto. A função deve retornar o valor do produto após a aplicação do desconto.
- 3) Faça uma função que receba o valor de um produto e um percentual de acréscimo. A função deve retornar o valor do produto após a aplicação do acréscimo.
- 4) Escreva um procedimento que recebe dois números inteiros e imprime, a soma, o produto, a diferença, o quociente e o resto entre esses dois números.



- 5) Considerando a fórmula para o cálculo da distância entre dois pontos (x1, y1) e (x2,y2):
  - a) Escreva uma função que receba como parâmetros as coordenadas de dois pontos e retorne a distância entre eles.
  - b) Escreva um programa C# (função principal) que capture do teclado as coordenadas dos 3 vértices de um triângulo, calcule e imprima o perímetro deste triângulo. Este programa deve utilizar a função do item anterior.



 Escopo: contexto que define a visibilidade e acessibilidade das variáveis em diferentes partes do programa.

 Toda variável tem um escopo. O escopo da variável equivale às linhas de código onde a variável pode ser acessada, lida e/ou modificada.



- São denominadas variáveis locais:
  - as variáveis declaradas na sub-rotina;
  - todos os parâmetros recebidos pela sub-rotina.
- O escopo de uma variável local corresponde apenas ao bloco de comandos de sua sub-rotina.
- Dentro de uma sub-rotina não se tem acesso a variáveis declaradas em outra sub-rotina.



- A reserva de memória para uma variável local é condicionada à execução da subrotina:
  - A reserva é refeita cada vez que a sub-rotina é executada (podem ser reservados endereços distintos a cada execução).
  - Quando a execução da sub-rotina termina, os espaços de memória reservados (e as respectivas variáveis associadas) são liberados para outros usos e já não podem ser acessados.



Os nomes das variáveis locais coincidentemente são iguais mas elas são completamente independentes.

O endereço de memória da variável raio da função main é diferente do endereço de memória do parâmetro raio da sub-rotina

```
static double VolumeCilindro (double raio, double altura)
  double volume = 3.14159 * raio * raio * altura;
  return volume:
static void Main(string[] args)
 double raio, altura, volume;
  Console.Write("Entre com o valor do raio: ");
  raio = Convert.ToDouble(Console.ReadLine());
 Console.Write ("Entre com o valor da altura: ");
  altura = Convert.ToDouble(Console.ReadLine());
 volume = VolumeCilindro(raio, altura);
  Console.Write("Volume do cilindro = ");
 Console.Write(volume);
```



A modificação do nome dos parâmetros e variáveis da função não implica em qualquer modificação no programa.

```
static double VolumeCilindro (double r, double h)
 ▼double v = 3.14159 * r * r * h;
  return v;
static void Main(string[] args)
 double raio, altura, volume;
  Console.Write ("Entre com o valor do raio: ");
  raio = Convert. To Double (Console. ReadLine());
  Console.Write ("Entre com o valor da altura: ");
  altura = Convert.ToDouble(Console.ReadLine());
  volume = VolumeCilindro(raio, altura);
  Console.Write("Volume do cilindro = ");
  Console.Write(volume);
```



Neste exemplo, quando a função é chamada, o valor das variáveis raio e altura declaradas na função main são utilizados para inicializar os parâmetros raio e altura da função VolumeCilindro.

```
static double VolumeCilindro (double raio, double altura)
 double volume = 3.14159 * raio * raio * altura;
 return volume;
static void Main(string[] args)
 double raio, altura, volume;
  Console.Write ("Entre com o valor do raio: ");
  raio = Convert.ToDouble(Console.ReadLine());
 Console.Write ("Entre com o valor da altura: ");
  altura = Convert.ToDouble(Console.ReadLine());
 volume = VolumeCilindro(raio, altura);
 Console.Write("Volume do cilindro = ");
 Console.Write(volume);
```



- Quando uma sub-rotina é chamada, é necessário indicar um valor de entrada para cada um de seus parâmetros.
- Este valor pode ser obtido diretamente de uma constante, de uma variável, de uma expressão ou, até mesmo, do valor de retorno de outra função.

```
volume = VolumeCilindro( 4.0, 5.0 );
volume = VolumeCilindro( raio, 10.0 / 2 );
volume = VolumeCilindro( diametro / 2, altura );
volume = VolumeCilindro( raio, Math.Sqrt( 9 ) + Math.Sqrt( raio ) );
```



 Quando uma única variável indica o valor de um parâmetro em uma chamada, apenas o valor desta é repassado para a sub-rotina e utilizado na inicialização do parâmetro.

```
static double VolumeCilindro (double r, double a)
{
  double v = 3.14159 * r * r * a;
  return v;
}
static void Main(string[] args)
{
  double raio=4.0, altura=5.0, volume;
  volume = VolumeCilindro(raio, altura);
  Console.Write("Volume do cilindro: %f", volume);
}
```



- Não há relação entre as variáveis raio e r.
- Se o valor de r for modificado na sub-rotina, a variável raio permanece inalterada.

```
static double VolumeCilindro (double r, double a)
{
  double v = 3.14159 * r * r * a;
  return v;
}
static void Main(string[] args)
{
  double raio=4.0, altura=5.0, volume,
  volume = VolumeCilindro(raio, altura);
  Console.Write("Volume do cilindro: %f", volume);
}
```



 Esta independência entre variável e parâmetro, indica a ocorrência de passagem de parâmetro por valor.

```
static double VolumeCilindro (double r, double a)
{
  double v = 3.14159 * r * r * a;
  return v;
}
static void Main(string[] args)
{
  double raio=4.0, altura=5.0, volume;
  volume = VolumeCilindro(raio, altura);
  Console.Write("Volume do cilindro: %f", volume);
}
```



- Existe outro tipo de relação entre variável e parâmetro denominado passagem de parâmetro por referência.
- Neste caso, o parâmetro da sub-rotina recebe um endereço de uma variável (ao invés de um valor de determinado tipo).



 Se houver alteração no local indicado pelo parâmetro que recebeu um endereço de variável, a respectiva variável será modificada, mesmo sem ter sido declarada na sub-rotina.



 A passagem de parâmetro por referência não será detalhada nesta disciplina.



- 6) Faça o teste de mesa dos programas gerados no exercício anterior.
- 7) Faça o teste de mesa do programa abaixo:

```
static int Calculo (int p, int q)
{
    p = p * 10;
    q = q + 10;
    return(p + q);
}

static void Main(string[] args)
{
    int x = 2, y = 5;
    Console.Write("{0} {1} {2}",x,y, Calculo(x,y));
}
```



## Sub-rotinas

#### Aula de Exercícios

Professor: Camillo Falcão





#### A forma geral de uma sub-rotina é:

```
TipoDeDados NomeDasub-rotina ( listaParametros )
{
  corpo da sub-rotina;
  return xxx;
}
```



#### Revisão: Tipos de sub-rotinas

#### Procedimentos

```
void ImprimeSoma (int a, int b)
{
   Console.Write(a+b);
}
```

#### Funções

```
int Soma(int a, int b)
{
  return (a+b);
}
```



# Implementação de sub-rotinas em C#

```
int Func1(int a)
{
    ...
    return val;
}

static void Main(string[] args)
{
    ...
    x=Func1(10);
    ...
}
```

```
static void Main(string[] args)
{
    ...
    x=Func1(10);
    ...
}
int Func1(int a)
{
    ...
    return val;
}
```



#### Exemplo completo - Procedimento

Escreva um programa que leia as três notas de um aluno. Calcule a média desse aluno e mostre, ao final, a média calculada.

O cálculo da média e a impressoes devem ser feitos por um procedimento.

```
static void CalculaMedia (double notal, double nota2, double nota3)
 double media;
 media = (nota1 + nota2 + nota3) / 3;
  Console.Write ("Media = \{0:N2\}", media);
static void Main(string[] args)
 double n1, n2, n3;
  Console.Write ("Digite as 3 notas: ");
 n1 = Convert.ToDouble(Console.ReadLine());
 n2 = Convert.ToDouble(Console.ReadLine());
 n3 = Convert.ToDouble(Console.ReadLine());
 CalculaMedia(n1, n2, n3);
```



#### Exemplo completo - Função

Escreva uma programa que calcule e imprima o Quadrado de um número. O cálculo deve ser feito por uma função.

```
static int Quadrado(int x)
  return (x * x);
static int Main(string args[])
  int numero, res;
  Console.Write ("Digite um numero inteiro: ");
  numero = Convert.ToInt32(Console.ReadLine());
  res = Quadrado (numero);
  Console.Write("O Quadrado de {0} é {1}.\n", numero, res);
```



- 1) Faça um procedimento que receba por parâmetro o tempo de duração de um experimento expresso em segundos e imprima na tela esse mesmo tempo em horas, minutos e segundos.
- 2) Faça uma função que receba por parâmetro o raio de uma esfera e calcule o seu volume: v = (4 \* PI \* R³) /3.
- 3) Faça uma função que receba a idade de uma pessoa em anos, meses e dias e retorne essa idade expressa em dias.



- 4) Faça uma função que receba dois números reais, a e b, e retorne o valor de (A<sup>2</sup> + B<sup>2</sup>)<sup>1/2.</sup>
- Observação: para calcular a raiz quadrada será preciso utilizar a função Math.Sqrt().
- 5) Considere as equações de movimento para calcular a posição (s) e velocidade (v) de uma partícula em um determinado instante t, dado sua aceleração a, posição inicial s0 e velocidade inicial v0, de acordo com as fórmulas:

$$s = s0 + v0*t + (a * t * t)/2$$
  
 $v = v0 + a*t$ 

Escreva um programa C# completo que capture os valores de s0, v0, a e t, fornecidos pelo usuário via teclado, e calcule e imprima os valores de s e v.



6) Considerando o critério de aprovação de uma disciplina que determina que um aluno está aprovado se a média ponderada de suas três provas for maior ou igual a 5.0, onde a média é dada pela fórmula:

Média = 
$$(P1 + P2 + 2.0 * P3) / 4.0$$

- (a) Escreva uma função que receba como parâmetros as notas das duas primeiras provas de um aluno (P1 e P2) e retorne a nota mínima que o aluno precisa tirar na terceira prova para que seja aprovado.
- (b) Escreva um programa em C# completo que leia do teclado as duas primeiras notas de um aluno, calcule e imprima a nota mínima que o aluno precisa tirar na P3 para que seja aprovado. Este programa deve fazer uso da função do item anterior.