



## **Aula: Introdução a Funções**

### **Introdução a Programação**

---

**Túlio Toffolo & Puca Huachi**  
<http://www.toffolo.com.br>

BCC201 – 2020/1  
Departamento de Computação – UFOP

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis
- 4 Valor de retorno
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis
- 4 Valor de retorno
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios

# O que é uma função?

É um conjunto de instruções desenhadas para cumprir determinada tarefa, agrupadas em uma unidade com um identificador para referi-la.

## Por que usar funções?

- **Reaproveitar** o código, ou seja, permitir que outras partes do programa ou outros programadores utilizem seus códigos;
- **Modularizar** um programa em partes menores;
- Aumentar a **legibilidade** e **manutenibilidade** do programa;
- O uso de funções geralmente diminui o tamanho do código.

## Exemplos de usos de funções

- Leitura de um número inteiro positivo;
- Imprimir um valor em um determinado formato;
- Cálculo do fatorial de um número;
- Encontrar o maior entre dois números.

**Na prática, qualquer sequência de instruções que apareça múltiplas vezes no código é candidata a ser uma função.**

## Exemplos de uso de funções

```
1 // função que calcula a raiz quadrada
2 double x = sqrt(y);
3
4 // função para gerar números aleatórios
5 int numero = rand();
6
7 // definição da função principal de um programa
8 int main() { ... }
```

**Bibliotecas C/C++ são compostas por funções, permitindo que o programador reaproveite códigos existentes.**

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções**
- 3 Escopo de variáveis
- 4 Valor de retorno
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios



# Protótipo de uma Função

Definição Geral do *Protótipo* de uma Função:

```
1 <tipo_retorno> <nome_função>(<lista_declaracao_parametro>);
```

Em que:

- **<tipo\_retorno>**: é o tipo do valor que a função retorna; quando a função não retorna nenhum valor utiliza-se a palavra chave **void**.
- **<nome\_função>**: é o identificador que nomeia a função.
- **<lista\_declaracao\_parametro>**: lista, possivelmente vazia, dos declarações separadas por vírgulas, dos parâmetros da função.

**Note que não é necessário definir os nomes dos parâmetros. É permitido incluir apenas os tipos no protótipo.**

# Implementação de uma Função

Implementação de uma Função em C:

```
1  <tipo_retorno> <nome_função>(<lista_declaração_parâmetro>) {  
2      <corpo_função>  
3  }
```

Em que:

- **<tipo\_retorno>**: tipo do valor que a função retorna; quando a função não retorna nenhum valor utiliza-se a palavra chave **void**.
- **<nome\_função>**: identificador que nomeia a função.
- **<lista\_declaração\_parâmetro>**: lista, possivelmente vazia, dos declarações separadas por vírgulas, dos parâmetros da função.
- **<corpo\_função>**: conteúdo (código fonte) da função.

# Protótipo de funções: exemplo

Tipo de retorno da função

Identificador/nome da função

Lista dos tipos dos parâmetros

(nome das variáveis é opcional)

```
void printReais(double);
```

# Implementação de funções: exemplo

Tipo de retorno da função

Identificador/nome da função

Lista de parâmetros

```
void printReais(double valor) {  
    printf("R$ %.2lf", valor);  
}
```

The diagram illustrates the components of the function definition `void printReais(double valor) { printf("R$ %.2lf", valor); }` using color-coded arrows and brackets:

- A red arrow points from the text "Tipo de retorno da função" to the `void` keyword.
- A blue arrow points from the text "Identificador/nome da função" to the `printReais` identifier.
- A green arrow points from the text "Lista de parâmetros" to the `(double valor)` parameter list.
- A large yellow bracket on the right side groups the opening curly brace, the function body, and the closing curly brace, with the label "Corpo da função" next to it.

## Mais exemplos: Conversão de Temperaturas

Fahrenheit e Celsius são duas escalas usadas para medir a temperatura.

- Desenvolveremos um programa para converter as temperaturas em Celsius para temperaturas equivalentes em Fahrenheit.
- A fórmula para conversão é:

$$F = 1.8 \times C + 32$$

- Onde  $C$  é a temperatura em Celsius e  $F$  é a temperatura correspondente em Fahrenheit.

## Exemplo: conversão de temperaturas

```
1  #include <stdio.h>
2
3  int main() {
4      double tempC, tempF;
5      printf("Conversão Celsius para Fahrenheit\n");
6      printf("(valor menor que -273.15 encerra o programa)\n\n");
7      printf("Temperatura em Celsius: ");
8      scanf("%lf", &tempC);
9
10     if (tempC >= -273.15) {
11         tempF = 1.8 * tempC + 32;
12         printf("%lf graus Celsius = %lf graus Fahrenheit.\n",
13             tempC, tempF);
14     }
15     return 0;
16 }
```

## Exemplo de execução

```
1  Conversão de Celsius para Fahrenheit  
2  (valor menor que -273.15 encerra o programa)  
3  
4  Temperatura em Celsius : 100  
5  100 graus Celsius = 212 graus Fahrenheit.
```

## Exemplo: conversão de temperaturas usando função

```
1  #include <stdio.h>
2
3  // protótipo da função
4  double celsiusToFahrenheit(double tempCels);
5
6  // método main (principal)
7  int main() {
8      double tempC, tempF;
9      printf("Conversão Celsius para Fahrenheit\n");
10     printf("(valor menor que -273.15 encerra o programa)\n\n");
11     printf("Temperatura em Celsius: ");
12     scanf("%lf", &tempC);
13
14     if (tempC >= -273.15) {
15         tempF = celsiusToFahrenheit(tempC);
16         printf("%lf graus Celsius = %lf graus Fahrenheit.\n",
17             tempC, tempF);
18     }
19     return 0;
20 }
```



## Exemplo: conversão de temperaturas usando função

```
1 // definição da função
2 double celsiusToFahrenheit(double tempCels) {
3     double f;
4     f = 1.8 * tempCels + 32;
5     return f;
6 }
```

OU

```
1 // definição da função
2 double celsiusToFahrenheit(double tempCels) {
3     return 1.8 * tempCels + 32;
4 }
```

## Exemplo: conversão de temperaturas usando função

Eliminando o protótipo da função:

```
1  #include <stdio.h>
2
3  // definição da função
4  double celsiusToFahrenheit(double tempCels) {
5      return 1.8 * tempCels + 32;
6  }
7
8  // O main fica idêntico ao do exemplo anterior
9  int main() {
10     ...
11 }
```

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis**
- 4 Valor de retorno
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios

# Escopo de variáveis

**As variáveis só existem no bloco onde foram declaradas.**

As variáveis `tempC` e `tempF` a seguir não podem ser usadas na função `celsiusToFahrenheit()`. De forma análoga, `tempCels` não pode ser usada na função `main()`. Essas variáveis são ditas **locais**.

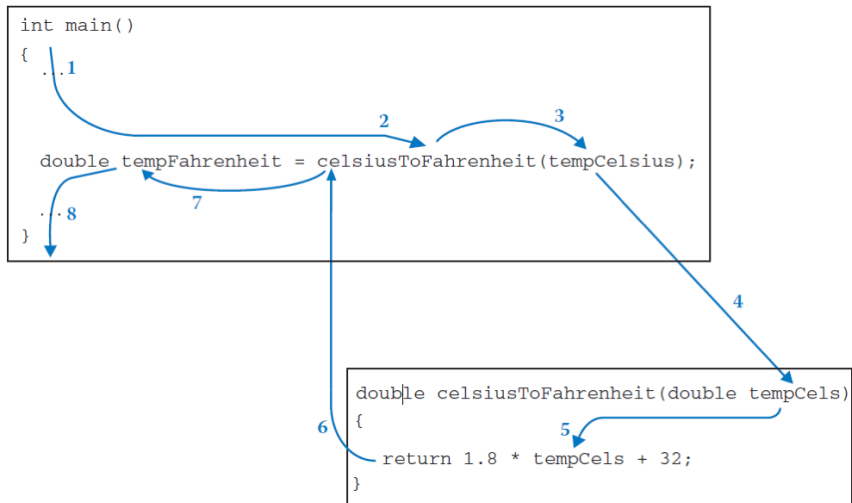
```
1 double celsiusToFahrenheit(double tempCels) {
2     return 1.8 * tempCels + 32;
3 }
4
5 int main() {
6     double tempC, tempF;
7     printf("Conversão Celsius para Fahrenheit\n");
8     printf("(valor menor que -273.15 encerra o programa)\n\n");
9     printf("Temperatura em Celsius: ");
10    scanf("%lf", &tempC);
11
12    if (tempC >= -273.15) {
13        tempF = celsiusToFahrenheit(tempC);
14        printf("%lf graus Celsius = %lf graus Fahrenheit.\n",
15              tempC, tempF);
16    }
17    return 0;
18 }
```

## Escopo de variáveis

**Obs. 1:** Mesmo que as variáveis possuam o mesmo nome na `main()` e na `celsiusToFahrenheit()`, que é uma declaração correta, o compilador enxerga como variáveis distintas.

**Obs. 2:** Pode-se declarar variáveis globais, para serem utilizadas em todo o programa. Porém, seu uso não é uma boa prática de programação, e deve ser restrito.

# Fluxo de execução



# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis
- 4 Valor de retorno**
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios



## Exemplo: Função que retorna valor

Função que recebe dois valores e retorna o maior valor.

```
1 // definição da função maior entre 2 números
2 int maior2(int a, int b) {
3     int maior;
4
5     if (a > b)
6         maior = a;
7     else
8         maior = b;
9
10    return maior;
11 }
```

## Exemplo: Função que retorna valor

Função que recebe três valores e retorna o maior valor.

```
1 // definição da função maior entre 3 números
2 int maior3(int a, int b, int c) {
3     int maior;
4
5     if ((a > b) && (a > c))
6         maior = a;
7     else {
8         if (b > c)
9             maior = b;
10        else
11            maior = c;
12    }
13    return maior;
14 }
```

## Exemplo: Função que não retorna valor

Função que recebe um parâmetro e não retornar nenhum valor.

```
1 // função que imprime um número como moeda
2 void printAsMoney(double n) {
3     printf("R$ %.2lf", n);
4 }
```

Uso:

```
1 int main() {
2     // ...
3     printAsMoney(salario);
4     // ...
5 }
```

## Exemplo: Função sem parâmetro e retorno

Função que **não** possui parâmetro e **não** retornar nenhum valor.

```
1 // Implementação da função que toca um beep
2 void beep(void) {
3     printf("\a");
4 }
```

Uso:

```
1 int main() {
2     ...
3     int x;
4     printf("Digite um número positivo: ");
5     scanf("%d", x);
6     if (x < 0)
7         beep();
8     ...
9 }
```

# Exemplo: Função com vários parâmetros

## Exemplos de protótipos das funções

```
1 // Verifica se os valores formam um triângulo
2 int ehTriangulo(int a, int b, int c);
3
4 // Calcula o valor da prestação de um produto
5 double valorPrestacao(double valor, double taxa, int numParcelas);
6
7 // Recebe 'F' ou 'M' e imprime o sexo por extenso:
8 // "Feminino" ou "Masculino"
9 void printSexo(char s);
```

## Exemplo: Uso de funções em argumentos

Chamadas a funções usadas com argumento de outras funções

```
1 // soma dois números
2 int soma (int m, int n) {
3     return m + n;
4 }
5
6 // Soma o quadrado de dois números
7 int somaQuadrado(int a, int b) {
8     return soma(pow(a, 2), pow(b, 2));
9 }
```

## Exemplos

Encontre o perímetro de um triângulo, dados os comprimentos de seus três lados. Use uma função para calcular o perímetro.

Obs.:  $P = a + b + c$ , onde  $a$ ,  $b$  e  $c$  são os lados do triângulo.

# Exemplos

```
1  #include <stdio.h>
2
3  float perimetro(float, float, float);
4
5  int main() {
6      float a, b, c;
7      printf("Digite os lados do triângulo: ");
8      scanf("%f %f %f", &a, &b, &c);
9
10     float p = perimetro(a, b, c);
11     printf("Perimetro = %f\n", p);
12     return 0;
13 }
14
15 float perimetro(float l1, float l2, float l3) {
16     return l1 + l2 + l3;
17 }
```



## Exemplos

Encontre a área de um triângulo, dados os comprimentos dos três lados.  
Utiliza a fórmula de Hero:

$$\sqrt{s(s-a)(s-b)(s-c)}$$

onde  $s$  é a metade do perímetro (use a função do exemplo anterior).

# Exemplos

```
1  #include <stdio.h>
2  #include <math.h>
3
4  float perimetro(float, float, float);
5  float areaTriangulo(float, float, float);
6
7  int main() {
8      float a, b, c;
9      printf("Digite os lados do triângulo: ");
10     scanf("%f %f %f", &a, &b, &c);
11
12     float area = areaTriangulo(a, b, c);
13     printf("Área = %f\n", area);
14 }
15
16 float areaTriangulo(float l1, float l2, float l3) {
17     float s = perimetro(l1, l2, l3) / 2.0;
18     return sqrt(s * (s-a) * (s-b) * (s-c));
19 }
```

## Exemplos

Crie uma função que retorna qual o conceito dada uma nota.  
Utilize a tabela a seguir:

Conceito	Nota
A	$9 \leq nota \leq 10$
B	$8 \leq nota < 9$
C	$7 \leq nota < 8$
D	$6 \leq nota < 7$
F	$nota < 6$

## Exemplos

```
1 // Função que recebe a nota e retorna qual o conceito ('A', 'B', etc.)
2 char conceito(double nota) {
3     if (9 <= nota && nota <= 10)
4         return 'A';
5     else if (8 <= nota)
6         return 'B';
7     else if (7 <= nota)
8         return 'C';
9     else if (6 <= nota)
10        return 'D';
11    else
12        return 'F';
13 }
```

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis
- 4 Valor de retorno
- 5 Passagem de parâmetro**
- 6 Exemplos e exercícios

# Passagem de Parâmetros

Os parâmetros formais (variáveis locais, declaradas como parâmetro da função chamada) são inicializados com o **valor** dos parâmetros.

- **Passagem por valor** – O valor dos parâmetros formais, **se** alterados durante a execução da função **não** acarretarão em nenhuma modificação no valor dos parâmetros reais (variáveis da função chamadora).

**Observação:** Todos os exemplos mostrados até o momento utilizam passagem de parâmetro **por valor**. De certa forma, podemos afirmar que em C sempre passamos um **valor** por parâmetro; até quando passamos um endereço de memória!

## Exemplo

Fazer uma função em C para trocar dois números.

- A função recebe dois valores e retorna esses valores trocados.
- Problema: Como retornar dois valores?

# Passagem de Parâmetro por Valor

Declaração da função:

```
1 void troca1 (int a, int b)
2 {
3     int temp = a;
4     a = b;
5     b = temp;
6 }
```

Chamada da função:

```
1 ...
2 c = 4; d = 5;
3 printf("c = %d, d = %d\n", c, d);
4 troca1(c, d);
5 printf("c = %d, d = %d\n", c, d);
6 ...
```



## Passagem de parâmetro

Saída do programa:

```
1  ...  
2  c = 4, d = 5  
3  c = 4, d = 5  
4  ...
```

- O programa passa os valores das variáveis  $c$  e  $d$  para as variáveis  $a$  e  $b$ , respectivamente.
- As variáveis possuem **escopo diferentes** e são **independentes**.
- Os valores de  $a$  e  $b$  são alterados, mas  $c$  e  $d$  permanecem os mesmos (**nada foi passado de volta para a unidade chamadora**).

# Como alterar o valor da variável dentro da função

Conversaremos sobre isso na **próxima aula!!!**

# Aula: Introdução a Funções

- 1 Definição de Função
- 2 Protótipo e algumas funções
- 3 Escopo de variáveis
- 4 Valor de retorno
- 5 Passagem de parâmetro
- 6 Exemplos e exercícios**

## Exemplos e exercícios

### Exemplo 1

Crie uma função que retorna 1 se o aluno for aprovado em uma disciplina e 0 caso contrário, considerando que as seguintes informações são passadas como argumentos:

- o número total de aulas de uma disciplina;
- o número de faltas do aluno (que deve ser  $\leq 25\%$  das aulas);
- a nota deste aluno (que deve ser  $\geq 6$ ).

Utilize o seguinte protótipo:

```
1 int aprovado(int, int, double);
```

## Exemplos e exercícios

### Exercício 1

Crie uma função que recebe a idade de uma pessoa e imprime a sua classe eleitoral, de acordo com a tabela abaixo:

Classe	Idade
Não-eleitor	Abaixo de 16 anos
Eleitor facultativo	Entre 16 e 18 anos e maior que 65 anos
Eleitor obrigatório	Entre 18 e 65 anos

### Exercício 2

Crie uma função que recebe um caractere e retorna o inteiro 1 se o caractere for uma letra minúscula (a-z), 2 se for maiúscula (A-Z) ou 0 se for outro caractere (!, @, #, \$, %, 1, 2, 3, etc).

Dica: lembre-se da tabela ASCII!



Perguntas?