



Introdução à linguagem C

Aula 04 - Funções

Professor: Racyus Delano

E-mail: racyuspacifico@univicosa.com.br

Funções

- **Funções definidas pelo programador**
 - Modulariza blocos de código;
 - Melhora manutenção e gerenciamento do código;
 - Melhora a legibilidade;
 - Fornece reusabilidade
 - As funções podem ser usadas várias vezes no mesmo programa;
 - Funções podem ser usadas em outros programas;

Funções

- **Componentes da função**

- Três passos para implementar funções

- 1) Declaração/protótipo da função;

- 2) Definição da função;

- 3) Chamada da função.

- 1 e 2 podem ser definidas:

- i. No mesmo arquivo que o main()

- ii. Em um arquivo separado para que outros programas possam utilizar

- `#include<arquivo.h>`

Funções

- **Declaração/protótipo da função**

- Uma declaração “informativa” para o compilador;
- Diz para o compilador como interpretar a chamada da função.

- **Sintaxe:**

<Tipo do retorno> Nome função (<Lista parâmetros>)

- **Sintaxe parâmetro:**

<Tipo do dado> nome do parâmetro

- **Exemplo:**

```
char grade(int score);
```

Funções

- **Declaração/protótipo da função**
 - Localizado antes de chamar a função;
 - Geralmente acima de todas as funções;
 - **Exemplo:**

```
#include <stdio.h>

// Function prototypes
double total_cost(int quantity, double unit_cost);

int main() {
```

Funções

- **Declaração/protótipo da função (Alternativa)**

- A declaração da função é um “informativo” para o compilador, então:
- Compilador apenas precisa conhecer:
 - Tipo de retorno;
 - Nome da função;
 - Lista de parâmetros.
- **Exemplo:**

```
#include <stdio.h>

// Function prototypes
double total_cost(int, double);

int main() {
```

Funções

- Definição da função

- Formato geral: cabeçalho e bloco de código:

<Tipo do retorno> Nome função (<Lista parâmetros>)

Bloco de código



Cabeçalho


- Exemplo:

```
double total_cost(int quantity, double unit_cost) {  
    const double TAXRATE = 0.05;  
    double sub_total;  
    sub_total = quantity * unit_cost;  
    return (sub_total + sub_total * TAXRATE);  
}
```

Funções

- Instrução de retorno
 - **Sintaxe:** `return <valor de retorno>`
 - Duas ações
 - Define o valor de retorno;
 - Transfere o controle de volta para função de chamada;
 - Boa prática de programação
 - Um retorno por função;
 - Return é a última instrução.

```
double total_cost(int quantity, double unit_cost) {  
    const double TAXRATE = 0.05;  
    double sub_total;  
    sub_total = quantity * unit_cost;  
    return (sub_total + sub_total * TAXRATE);  
}
```



Funções

- **Chamada da função**

- Usar o nome da função transfere o controle para a função

1) Os valores são passados por meio de parâmetros;

2) As instruções dentro da função são executadas;

3) O controle continua após a chamada.

- Para funções que retornam valor,

- Armazene o valor para uso posterior

```
bill = total_cost(number, price);
```

- Use o valor retornado sem armazenar

```
printf("Cost is %f\n", total_cost(number, price));
```

- Descarte o valor de retorno

```
total_cost(number, price);
```

Funções

- **Declarando funções void**

- Semelhante às funções que retornam um valor.
- No entanto, nenhum valor é retornado;
- Funções void não tem instrução return;
- **Exemplo:**

```
void showResults(double fDegrees, double cDegrees) {  
    printf("%.2f degrees fahrenheit equals ", fDegrees);  
    printf("%.2f degrees celsius\n", cDegrees);  
}
```

Funções

- Chamando funções void

- De outra função, por exemplo, main():

```
showResults(degreesF, degreesC);  
showResults(32.5, 0.3);
```

- Não pode ser usado onde um valor é necessário;
 - Não pode ser atribuído a uma variável, já que não retorna nenhum valor.

Funções

- **Documentação função**

- Usado para auxiliar na manutenção do programa;
- Comentários abaixo do cabeçalho
 - Finalidade da função
 - Parâmetros
 - Retorno
- **Exemplo:**

```
double interest(double balance, double rate);  
// Calculates the interest charge on an account balance  
// Parameters:   balance - non-negative account balance  
//              rate   - interest rate percentage  
// Return:      calculated interest charge
```

Funções

- **main(): ‘Função especial’**
 - É uma função;
 - É a primeira função executada;
 - Chamada pelo sistema operacional;
 - Retorna um valor para o sistema operacional;
 - Valor retornado pode ser testado por scripts.
 - Zero indica o fim do programa.

Exercícios sala de aula

1. Escreva um programa que solicite dois números ao usuário e apresente na tela o resultado da soma desses números. Esse programa deve possuir uma função para calcular a soma.
2. Escreva um programa que solicite dois números do tipo inteiro distintos ao usuário e que apresente na tela o maior deles. Esse programa deve possuir uma função para verificar qual é o maior número.

Exercícios da lista 01

- **Data de entrega: 05/03/2024.**

1. Escreva um programa que solicite dois números inteiros ao usuário e que apresente na tela como resultado o dobro dos números que devem ser somados e o resultado da soma devem ser triplicados. Esse programa deve possuir uma função para dobrar o valor de um número, outra para somar dois números e uma terceira para triplicar um número.
2. Desenvolva um programa que tem uma função que recebe um valor inteiro e verifique se o valor é positivo ou negativo. A função deve retornar um valor booleano. Depois a função principal deve informar ao usuário o resultado.