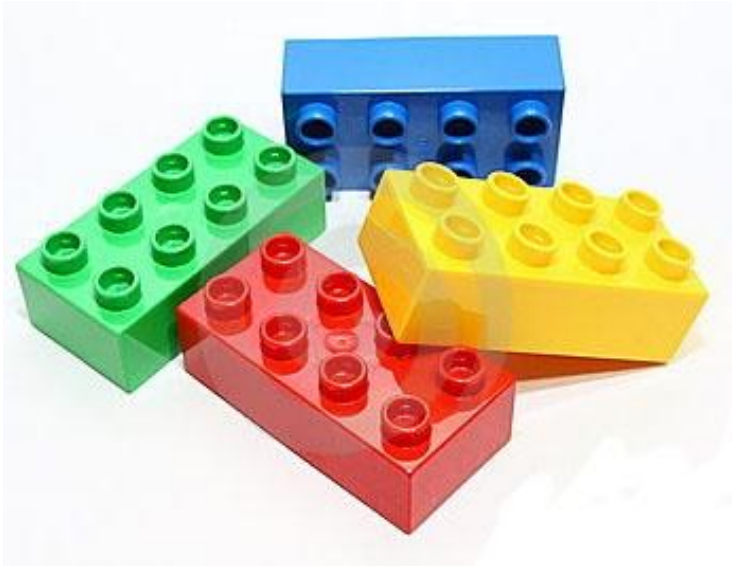
Funções

Laboratório de Programação

Prof. André Bezerra

O que são funções?

Conjunto de comandos agrupados em um **bloco** que recebe um **nome** e através deste pode ser ativado.



Porque usar funções ?

- Para permitir o reaproveitamento de código já construído(por você ou por outros programadores);
- Para evitar que um trecho de código que seja repetido várias vezes dentro de um mesmo programa;
- Para permitir a alteração de um trecho de código de uma forma mais rápida. Com o uso de uma função é preciso alterar apenas **dentro** da função que se deseja;
- Para que os blocos do programa não fiquem grandes demais e, por conseqüência, mais difíceis de entender;
- Para facilitar a leitura do programa-fonte de uma forma mais fácil;
- Para separar o programa em partes(blocos) que possam ser logicamente compreendidos de forma isolada.

Formato geral de uma função em C

```
tipo nome ([parametros]){
}

/*Exemplos*/

int soma(int x, int y){
  return x + y;
}

void imprime_boas_vindas(){
  printf("Ola, bem vindo!!");
}
```

Exemplo 01

```c
#include <stdio.h>

int soma( int x, int y){
   return x + y;
}

int main(){
   int a = 10, b = 20;
   printf("Valor: %d", soma(a,b));
  return 0;
}
```

```
SAÍDA
Valor : 30
```

Exemplo 02

```c
#include <stdio.h>
int soma( int x, int y){
   return x + y;
}
void imprimeValor( int x){
   printf("Valor: %d", x);
}

int main(){
   int a = 10, b = 20;
   imprimeValor( soma(a,b));
  return 0;
}
```

```
SAÍDA
Valor : 30
```

Escopo de Variáveis

Por **escopo de uma variável** entende-se o bloco de código onde esta variável é válida. Com base nisto, temos as seguintes afirmações:

- As variáveis valem no bloco que são definidas;
- As variáveis definidas dentro de uma função recebem o nome de **variáveis locais**;
- Os parâmetros formais de uma função valem também somente dentro da função;
- Uma variável definida dentro de uma função não é acessível em outras funções, mesmo que estas variáveis tenham nomes idênticos.

Exemplo 03

```c
#include <stdio.h>

void funcao01(){
    int x = 10;
    printf("X dentro da funcao 01: %d\n", x);
}
void funcao02(){
    int x = 12;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01();
    printf("X na funcao principal: %d\n", x);
    funcao02();
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

```
X na função principal: 15
X dentro da funcao 01: 10
X na função principal: 15
X dentro da funcao 02: 12
X na função principal: 15
```

Exemplo 04

```c
#include <stdio.h>

void funcao01( int x){
    x += 10;
    printf("X dentro da funcao 01: %d\n", x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

?

Exemplo 04

```c
#include <stdio.h>

void funcao01( int x){
    x += 10;
    printf("X dentro da funcao 01: %d\n", x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

```
SAÍDA
X na função principal: 15
X dentro da funcao 01: 25
X na função principal: 15
X dentro da funcao 02: 10
X na função principal: 15
```

Exemplo 05

```c
#include <stdio.h>

void funcao01( int *x){
    *x += 10;
    printf("X dentro da funcao 01: %d\n", *x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(&x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

?

Exemplo 05

```c
#include <stdio.h>

void funcao01( int *x){
    *x += 10;
    printf("X dentro da funcao 01: %d\n", *x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

```
X na função principal: 15
X dentro da funcao 01: 25
X na função principal: 25
X dentro da funcao 02: 20
X na função principal: 25
```

Exemplo 06

```c
#include <stdio.h>

void funcao01( int *x){
    *x += 10;
    printf("X dentro da funcao 01: %d\n", *x);
    funcao02(*x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(&x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

?

Exemplo 06

```c
#include <stdio.h>

void funcao01( int *x){
    *x += 10;
    printf("X dentro da funcao 01: %d\n", *x);
    funcao02(*x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

```
SAÍDA
X na função principal: 15
X dentro da funcao 01: 25
X dentro da funcao 02: 20
X na função principal: 25
X dentro da funcao 02: 20
X na função principal: 25
```

E agora? O que acontece ao executar este código?

```c
#include <stdio.h>
void funcao01( int *x){
    *x += 10;
    printf("X dentro da funcao 01: %d\n", *x);
    funcao02(*x);
}
void funcao02( int x){
    x -= 5;
    printf("X dentro da funcao 02: %d\n", x);
    funcao01(&x);
}
int main(){
    int x = 15;
    printf("X na funcao principal: %d\n", x);
    funcao01(&x);
    printf("X na funcao principal: %d\n", x);
    funcao02(x);
    printf("X na funcao principal: %d\n", x);
     return 0;
}
```

SAÍDA

?

LOOP ETERNO!!

Exercício 01

- Escreva uma função que receba como parâmetro o peso e a altura de uma pessoa. A função calcula e informa o **IMC**.

  O IMC é dado pela fórmula:

  IMC = PESO / (ALTURA * ALTURA)

Resposta Exercício 01

```c
#include <stdio.h>

void imc( float p, float a);

int main(){
  float peso, altura;

  scanf("%f", &peso);
  scanf("%f", &altura);

  imc(peso, altura);
  return 0;
}

void imc( float p, float a){
   float calculoImc = p / (a * a);

   printf("O IMC eh: %.2f", calculoImc);
}
```

Exercício 02

- Escreva uma função que receba dois números inteiros e informe o resultado da soma do primeiro numero multiplicado por dois, com o segundo número multiplicado por 4.

Resposta Exercício 02

```c
#include <stdio.h>

void soma(int a, int b);
int main(){
   int x, y;
   printf("Digite os valores de X e Y:");
   scanf("%d %d", &x, &y);
   soma(x,y);
   return 0;
}
void soma(int a, int b){
   int calculo;
   calculo = a * 2 + b * 4;
   printf("O resultado eh: %d", calculo);
}
```

Exercício 03

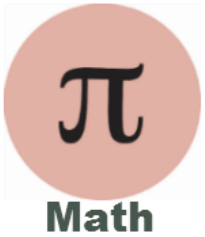
- Escreva uma função que recebe duas variáveis com valores inteiros por parâmetro. A função troca os valores destas variáveis, ou seja, a primeira variável ficará com o valor da segunda, e vice-versa.
- Imprima os valores antes e após a chamada da função.

Resposta Exercício 03

```c
#include <stdio.h>
void trocaValores(int *x, int *y);
int main(){
    int a, b;
    printf("Digite os valores para A e B: ");
    scanf("%d %d", &a, &b);
    printf("Valor de A: %d e B: %d\n", a, b);
    trocaValores(&a, &b);
    printf("Apos a funcao, valores trocados\n");
    printf("Valor de A: %d e B: %d\n", a, b);
    return 0;
}
void trocaValores(int *x, int *y){
    int aux = *x;
    *x = *y;
    *y = aux;
}
```

math.h

- É um arquivo de cabeçalho que fornece protótipos para funções matemáticas básicas.
- Contém funções trigonométricas, função para cálculo de raiz quadrada(número real), logaritmo, entre outras.



Funções trigonométricas

```
double sin( double);
double cos( double);
double tan( double);
```

Veremos exemplos desta funções nas próximas aulas.

Raiz e potência

```
double sqrt( double);
/* Uso típico: y = sqrt( x);
Devolve a raiz quadrada de x. Não use com x < 0. */


#include <stdio.h>
#include <math.h>

int main(){
   double x = 25.00;
   printf(“Raiz quadrada de %.2lf = %.2lf", x,
   sqrt(x));
   return 0;
}
```

```
SAÍDA
Raiz quadrada de 25.00 = 5.00
```

Raiz e potência

```
double pow( double, double);
/* Uso típico: p = pow( x, y);
Devolve x^y, ou seja, x elevado à potência y. */

#include <stdio.h>
#include <math.h>

int main(){
   double x = 2.0, y = 4.0;
   printf("%.2lf elevado a %.2lf = %.2lf", x, y,
   pow(x,y));
   return 0;
}
```

```
SAÍDA
2.00 elevado a 4.00 = 16.00
```

Arredondamentos

```
double floor( double);

/* Uso típico: i = floor( x). A função devolve o maior
inteiro que seja menor que ou igual a x, isto é,
o único inteiro i que satisfaz i <= x < i+1.          */

#include <stdio.h>
#include <stdlib.h>

int main(){
   double n = 25.89;
   printf("%lf", floor(n));
  return 0;
}
```

```
SAÍDA
26.00000
```

Arredondamentos

```
double ceil( double);

/* Uso típico: i = ceil( x). A função devolve o maior
inteiro que seja menor que ou igual a x, isto é,
o único inteiro i que satisfaz i <= x < i+1.          */

#include <stdio.h>
#include <stdlib.h>

int main(){
   double n = 25.89;
   printf("%lf", ceil(n));
  return 0;
}
```

```
SAÍDA
25.00000
```

Exercício 04

- Escreva uma função que receba um número real. A função imprime apenas a parte decimal do número.

Resposta Exercício 04

```c
#include <stdio.h>
#include <math.h>

void parteDecimal(float num){
   printf("%.2f", num - floor(num));
}

int main(){
   float n;
   scanf("%f", &n);
   parteDecimal(n);
   return 0;
}
```

Exercício 05

- Crie uma função que receba o nome de uma pessoa. Imprima dentro da função uma mensagem de **boas-vindas** com o nome passado por parâmetro.

Resposta Exercício 05

```c
#include <stdio.h>

void boasVindas(char s[20]){
   printf("Ola %s, bem vindo ao sistema!", s);
}

int main(){
   char nome[20];

   printf("Nome: ");
   scanf("%s", nome);

   boasVindas(nome);

  return 0;
}
```

Na próxima aula..

- Estrutura de Seleção em C
- Outras bibliotecas de funções
- Lista de Exercício 01

- Para ler
  - Livro: Linguagem C Descomplicada
    - Capítulos 02 e 03 (pág 32 a 94)