Universidade Federal do Piauí
Centro de Ensino Aberto e a Distância
Curso de Sistemas de Informação

Algoritmos e Programação I

Funções

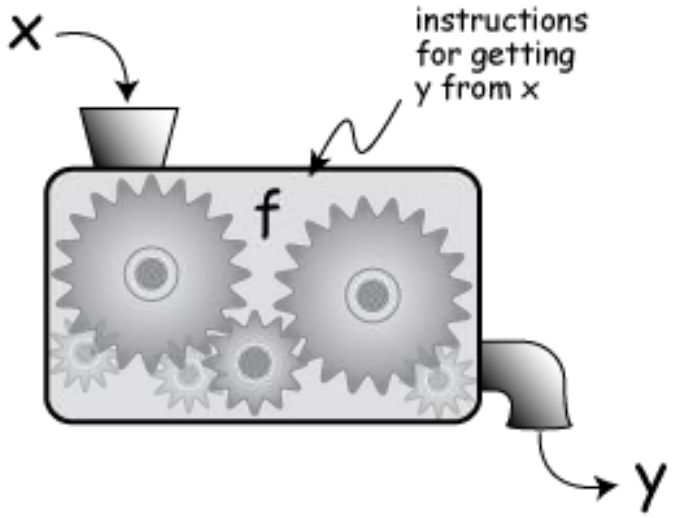
Prof. Arlino Magalhães
arlino@ufpi.edu.br

Sumário

Conceito

Função é um trecho de um algoritmo independente com um objetivo determinado, simplificando o entendimento do algoritmo e proporcionando ao algoritmo menos chances de erro e de complexidade.

Através da passagem de argumentos aos seus parâmetros e através de seu nome, trecho de código de uma função é executado.



Principais Objetivos das Funções

Dividir e estruturar um algoritmo em partes logicamente coerentes.

Facilitar o teste de trechos de código em separado.

Proporcionar ao programador a reutilização de código, através da criação de bibliotecas de funções personalizadas.

Maior legibilidade do algoritmo.

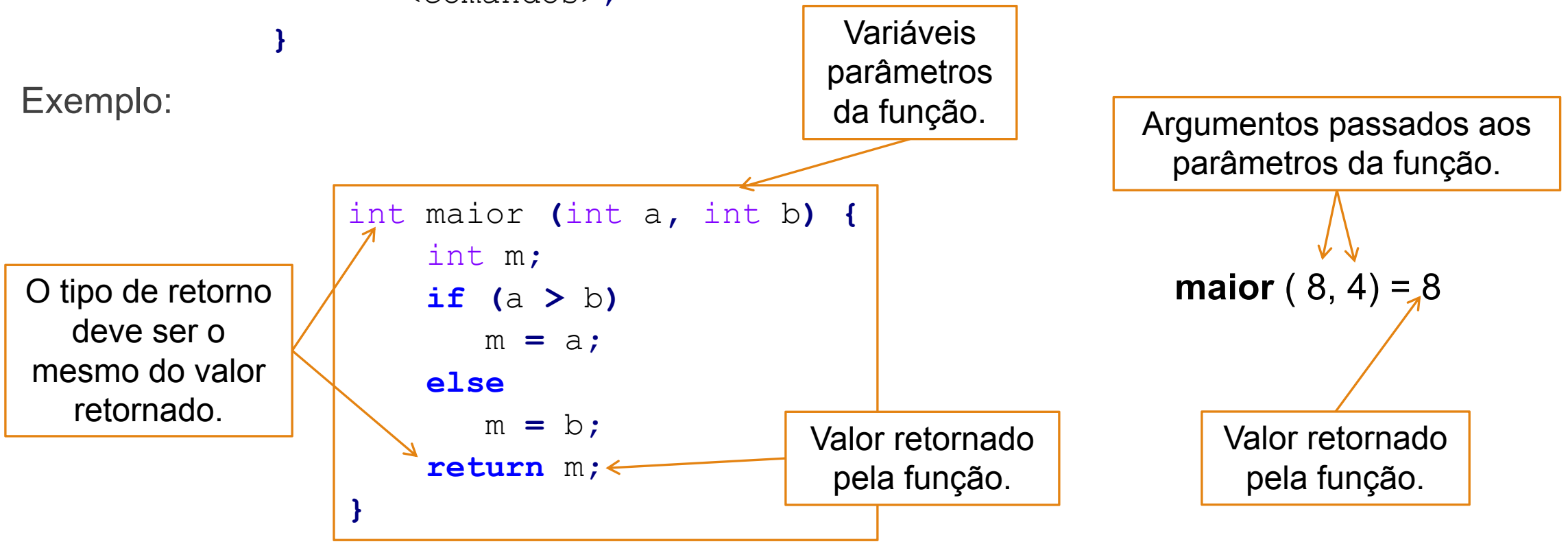
Evitar a repetição de código, substituindo códigos semelhantes por chamadas a uma única função.

Função

Sintaxe:

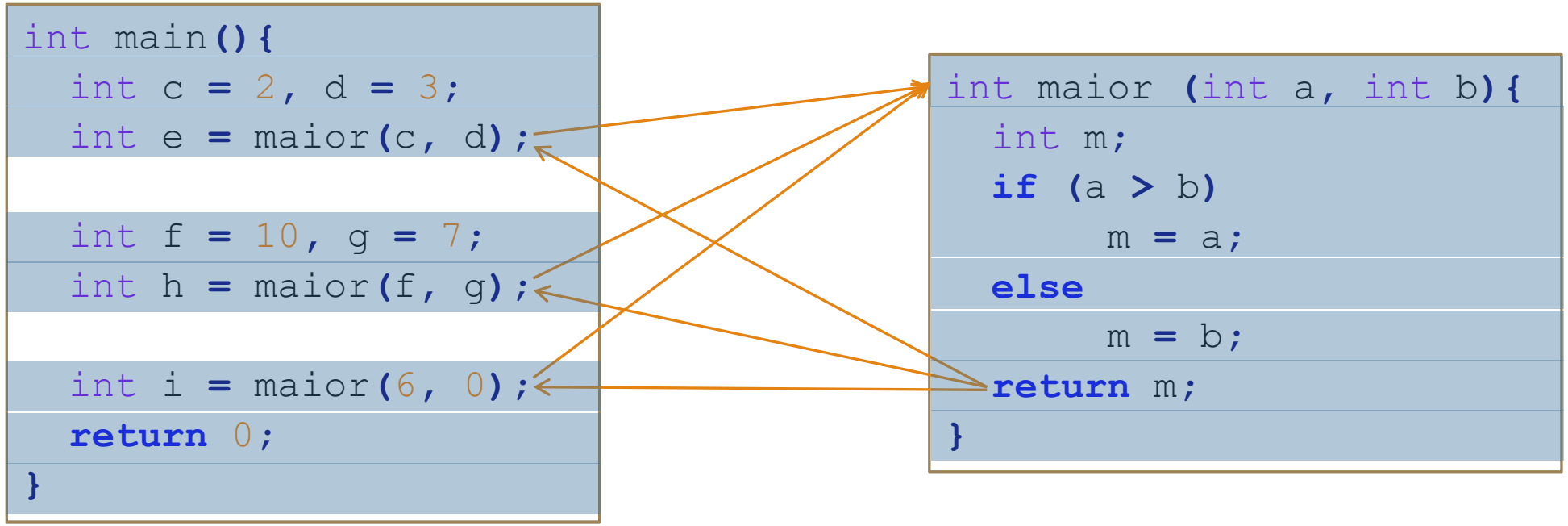
```
<tipo_de_retorno> <nome_da_função> (<lista_de_parâmetros>) {
<comandos>;
}
```

Exemplo:



Função - Chamada de Função

Quando uma função é chamada, o fluxo de controle do programa é desviado para a função. Ao terminar o execução dos comando da função, o fluxo de controle retorna ao comando seguinte àquele onde a função foi ativada.



Função - Função Maior
(Código Completo)

```c
#include <stdio.h>
int maior (int a, int b){
      int m;
      if (a < b)
            m = a;
      else
            m = b;
      return m;
}
int main (){
int c, d, e, f, g, h, i;
      c = 2;  d = 3;
      e = maior (c, d);
      f = 10; g = 7;
      h = maior (f, g);
      i = maior (6, 0);
      return 0;
}
```

Função - Exemplo

```c
#include <stdio.h>
int n1 = 3,  f1,  f2;
int fatorial (int num) {
     int fat = 1;
     for (int i = num;  i > 1;  i--)
          fat *= i;
     return fat;
}
int main (){
     f1 = fatorial (n1);
     printf ("%d \n", f1);
     f2 = fatorial (4);
     printf ("%d \n", f2);
     printf ("%d \n", fatorial (5));
     return 0;
}
```

Passagem de Parâmetros

Passagem de parâmetros por valor:

- Os parâmetros da função recebem uma cópia do valor das variáveis argumento quando a função é invocada.
- Todas as alterações feitas nos parâmetros dentro da função não afetam os valores originais dos argumentos.

Passagem de parâmetros por referência:

- Neste caso, é enviado para os parâmetros da função uma referência (endereço de memória) das variáveis argumento utilizadas, e não uma simples cópia.
- As alterações realizadas dentro da função nos parâmetros alteram os valores originais das variáveis argumento.

Passagem de Parâmetros

Passagem por Valor


```c
#include <stdio.h>
int c, d;
void troca (int a, int b){
      int aux;
      aux = a;
      a = b;
      b = aux;
}
int main (){
      c = 2;  d = 3;
      printf ("Antes: C=%d, D=%d \n", c, d);
      troca(c, d);
      printf ("Depois: C=%d, D=%d \n", c, d);
      return 0;
}
```

Passagem de Parâmetros

Passagem por Referência


```c
#include <stdio.h>
int c, d;
void troca (int *a, int *b){
      int aux;
      aux = a;
      a = b;
      b = aux;
}
int main (){
      c = 2;  d = 3;
      printf ("Antes: C=%d, D=%d \n", c, d);
      troca(&c, &d);
      printf ("Depois: C=%d, D=%d \n", c, d);
      return 0;
}
```

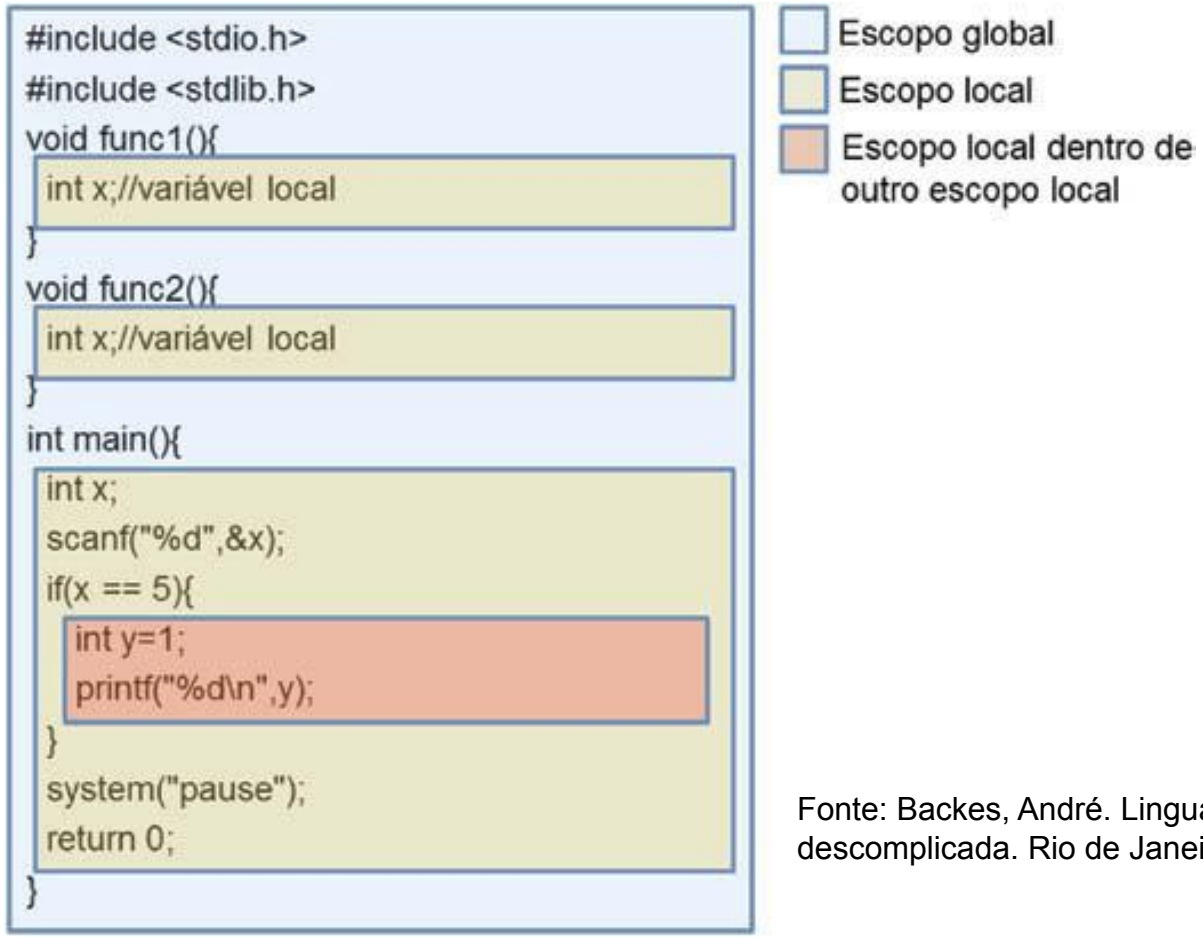
Escopo de Variáveis e Funções

O escopo é o conjunto de regras que determinam o uso e a validade de variáveis nas diversas partes do programa.

Principais tipos de escopos:

- Variáveis locais
  - São declaradas em um bloco (como uma função, por exemplo) e só tem validade dentro desse bloco.
- Parâmetros formais
  - São declarados como sendo as entradas (parâmetros) de uma função e só tem validade dentro dessa função.
- Variáveis globais
  - São declaradas fora de todas as funções do programa e tem validade em todas as funções.

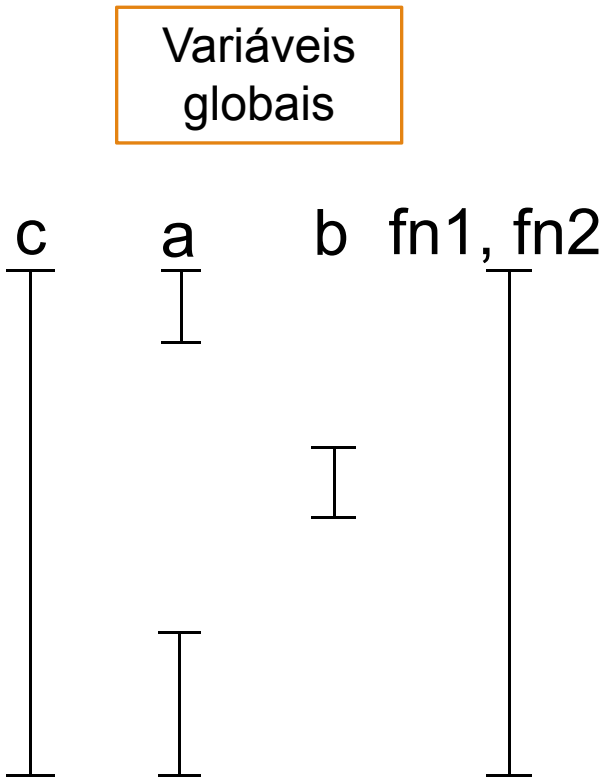
Escopo de Variáveis e Funções
Exemplo



Fonte: Backes, André. Linguagem C: completa e descomplicada. Rio de Janeiro: Elsevier, 2013

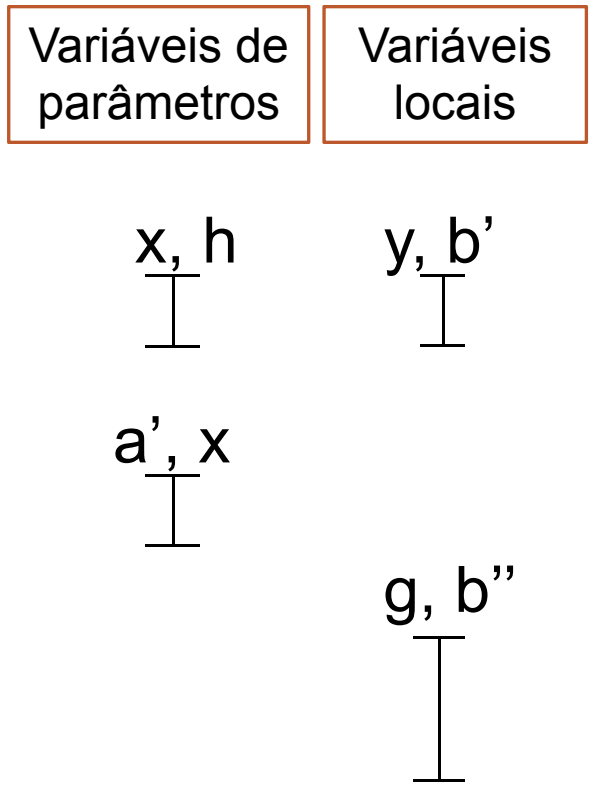
Exercício 01

Determinar o escopo das variáveis abaixo:



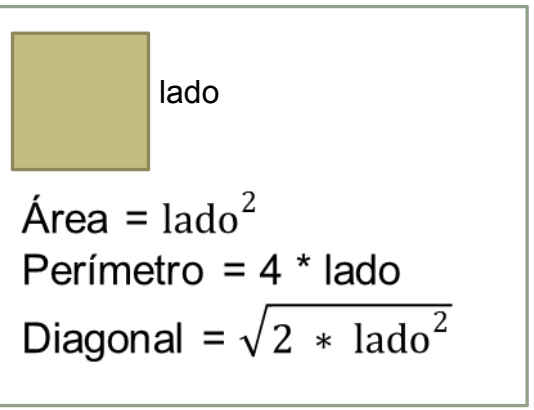

```
int c, a, b;
void fn1 (int  x, int h){
     int y, b;
}
void fn2 (int  a, int  x){

}
int main (){
     int g, b;
     return 0;
}
```



Exemplo

Operações em um Quadrado




```c
#include <stdio.h>
#include <math.h>
float area (float lado){
      return lado*lado;
}
float perimetro (float lado){
      return 4*lado;
}
float semiPerimetro (float lado){
      return perimetro (lado)/2;
}
float diagonal (float lado){
      float d = 2*lado*lado;
      return sqrt (d);
}
int main (){
    float lado = 4;
    printf ("Area=f%", area (lado));
    printf ("Perimetro=f%", perimetro (lado));
    printf ("Semi perimetro=f%", semiPerimetro (lado));
    printf ("Diagonal=f%", diagonal (lado));
    return 0;
}
```

Exercício 02

Fazer uma função chamada de **Igual** que receba dois números como parâmetros. A função retorna verdadeiro se os dois números forem iguais, caso contrário, retorna falso.

```
#include <stdio.h>
int Igual (int n1, int n2){
      int valor;
      if (n1 == n2)
            valor = 1 ;
      else
            valor = 0;
      return valor;
}
int main (){
      a = 2;  b = 3;
      if ( Igual (a, b) )
            printf ("Numeros iguais!");
      else
            printf ("Numeros diferentes!");
      return 0;
}
```

Exercício 02
(Resposta Melhorada)

Fazer uma função chamada de **Igual** que receba dois números como parâmetros. A função retorna verdadeiro se os dois números forem iguais, caso contrário, retorna falso.

```
#include <stdio.h>
int Igual (int n1, int n2){
      if (n1 == n2)
            return 1 ;
      else
            return 0;
}
int main (){
      a = 2;  b = 3;
      if ( Igual (a, b) )
            printf ("Numeros iguais!");
      else
            printf ("Numeros diferentes!");
      return 0;
}
```

Exercício 03

Fazer uma função chamada **Compara** que recebe dois números como parâmetros. A função deve retornar os seguintes resultados:

- 0, se os dois números forem iguais;
- 1, se o primeiro número for maior que o segundo;
- -1, se o segundo número for maior que o primeiro.

```
#include <stdio.h>
int Compara (int n1, int n2){
      if (n1 == n2)
            return 0;
      else
            if (n1 > n2)
                  return 1;
            else
                  return -1;
}
int main (){
      a = 2;  b = 3;
      switch ( Compara (a, b) ) {
            case 0: printf ("a = b"); break;
            case 1: printf ("a > b"); break;
            case -1: printf ("a < b"); break;
      }
      return 0;
}
```

Exercício 04

Fazer uma função que receba como parâmetro as três notas de um aluno e retorne o conceito de sua média conforme a tabela abaixo:

Média	Conceito
8,0 ≤ média ≤ 10,0	A
6,0 ≤ média < 8,0	B
4,0 ≤ média < 6,0	C
média < 4,0	D

```
char media (float n1, float n2, float n3) {
      float media = (n1 + n2 + n3) / 3;
      if (media > =  8)
            return 'A';
      else
            if (media >= 6)
                  return 'B';
            else
                  if (media >= 4)
                        return 'C';
                  else
                        return 'D';
}
```