Programação em C

Prof. Eduardo Magalhães

Programação em C

- LINGUAGEM C

Aula 02

Introdução às funções

- Uma função é um bloco de código de programa que pode ser usado diversas vezes em sua execução. O uso de funções permite que o programa fique mais legível, melhor estruturado. Um programa em C consiste, no fundo, de várias funções colocadas juntas.

Introdução às funções

- **Forma geral**

Apresentamos aqui a forma geral de uma função:

```
tipo_de_retorno nome_da_função (lista_de_argumentos)
{
    código_da_função
}
```

Introdução às funções

```
#include <stdio.h>
 int mensagem ()  /* Funcao simples: só imprime Olá! */
 {
   printf ("Olá! ");
   return(0);
}
   int main ()   {
   mensagem();
   printf ("Eu estou vivo!\n");
   return(0);
 }
```

Introdução às funções

- O programa defini uma função **mensagem()** que coloca uma string na tela e retorna 0. Esta função é chamada a partir de **main()**, que, como já vimos, também é uma função. A diferença fundamental entre main e as demais funções do problema é que main é uma função especial, cujo diferencial é o fato de ser a primeira função a ser executada em um programa.

Introdução às funções

- ARGUMENTOS
- são as entradas que a função recebe. É através dos argumentos que passamos *parâmetros* para a função. Já vimos funções com argumentos. As funções **printf()** e **scanf()** são funções que recebem argumentos. Vamos ver um outro exemplo simples de função com argumentos:

Introdução às funções

```
#include <stdio.h>
 int square (int x) /* Calcula o quadrado de x */     {
    printf ("O quadrado e %d",(x*x));
    return(0);
 }
    int main ()     {
    int num;
    printf ("Entre com um numero: ");
    scanf ("%d",&num);
    printf ("\n\n");
    square(num);
    return(0);
}
```

Introdução às funções

- Na definição de **square()** dizemos que a função receberá um argumento inteiro **x**. Quando fazemos a chamada à função, o inteiro **num** é passado como argumento. Há alguns pontos a observar. Em primeiro lugar temos de satisfazer aos requisitos da função quanto ao tipo e à quantidade de argumentos quando a chamamos.

Introdução às funções

- Em segundo lugar, não é importante o nome da variável que se passa como argumento, ou seja, a variável **num**, ao ser passada como argumento para **square()** é copiada para a variável **x**. Dentro de **square()** trabalha-se apenas com **x**. Se mudarmos o valor de x dentro de square() o valor de num na função main() permanece inalterado.

Introdução às funções

- Função de mais de uma variável:
  - neste caso, os argumentos são separados por vírgula e que deve-se explicitar o tipo de cada um dos argumentos, um a um. Note, também, que os argumentos passados para a função não necessitam ser todos variáveis porque mesmo sendo constantes serão copiados para a variável de entrada da função.

Introdução às funções

```
#include <stdio.h>
   int mult (float a, float b,float c) /* Multiplica 3 numeros */      {
   printf ("%f",a*b*c);
   return(0);      }
   int main ()      {
   float x,y;
   x=23.5;
   y=12.9;
   mult (x,y,3.87);
   return(0);
  }
```

Introdução às funções

- **Retornando valores**

  Às vezes é necessário fazer com que uma função retorne um valor. As funções que vimos até aqui estavam retornando o número 0. Podemos especificar um tipo de retorno indicando-o antes do nome da função.

Introdução às funções

- Mas para dizer ao C *o que* vamos retornar precisamos da palavra reservada **return**. Sabendo disto fica fácil fazer uma função para multiplicar dois inteiros e que retorna o resultado da multiplicação.

Introdução às funções

```
#include <stdio.h>
        int prod (int x,int y)        {
        return (x*y);
}
        int main ()   {
        int saida;
        saida=prod (12,7);
        printf ("A saida e: %d\n",saida);
        return(0);
 }
```

Introdução às funções

- Veja que, como prod retorna o valor de 12 multiplicado por 7, este valor pode ser usado em uma expressão qualquer. No programa fizemos a atribuição deste resultado à variável saida, que posteriormente foi impressa usando o printf.

Introdução às funções

- Uma observação adicional: se não especificarmos o tipo de retorno de uma função, o compilador C automaticamente suporá que este tipo é inteiro. Porém, não é uma boa prática não se especificar o valor de retorno e, neste curso, este valor será sempre especificado.

Introdução às funções

- Com relação à função main, o retorno sempre será inteiro. Normalmente faremos a função main retornar um zero quando ela é executada sem qualquer tipo de erro.
- Mais um exemplo de função, que agora recebe dois floats e também retorna um float:

Introdução às funções

```
#include <stdio.h>
float prod (float x,float y)   {
return (x*y);
 }
int main () {
float saida;
saida=prod (45.2,0.0067);
printf ("A saida e: %f\n",saida);
return(0);
}
```

Exercício Proposto

Escreva uma função que some dois inteiros e retorne o valor da soma.