

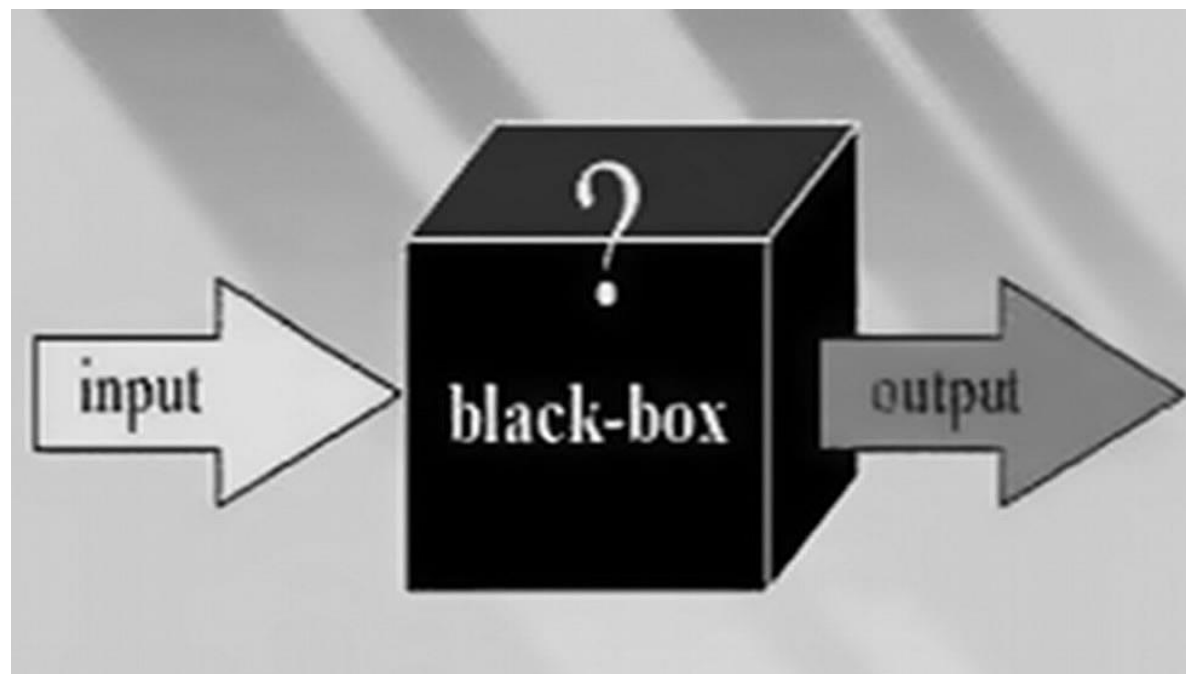


# Funções em C

**Profa. Elloá B. Guedes**  
[www.elloaguedes.com](http://www.elloaguedes.com)

# Utilização de Funções

- Já estamos utilizando funções em nossos programas
  - printf, scanf, getchar, etc.
- Biblioteca: stdio.h



# Exemplo

- Escreva um programa em C que coloque na tela a seguinte saída, escrevendo a linha de 20 asteriscos por meio de um laço for

```
*****  
Números entre 1 e 5  
*****  
1  
2  
3  
4  
5  
*****
```

# Funções

- Observe que o trecho para escrever 20 asteriscos na tela é repetido 3 vezes
- Função desse trecho de código: escrever uma linha de asteriscos na tela
  - Vamos escrever uma função chamada “linha” com esta tarefa?
- Vantagens
  - Evita repetição de código
  - Permite fácil alteração ou correção de erros

# Funções - Características

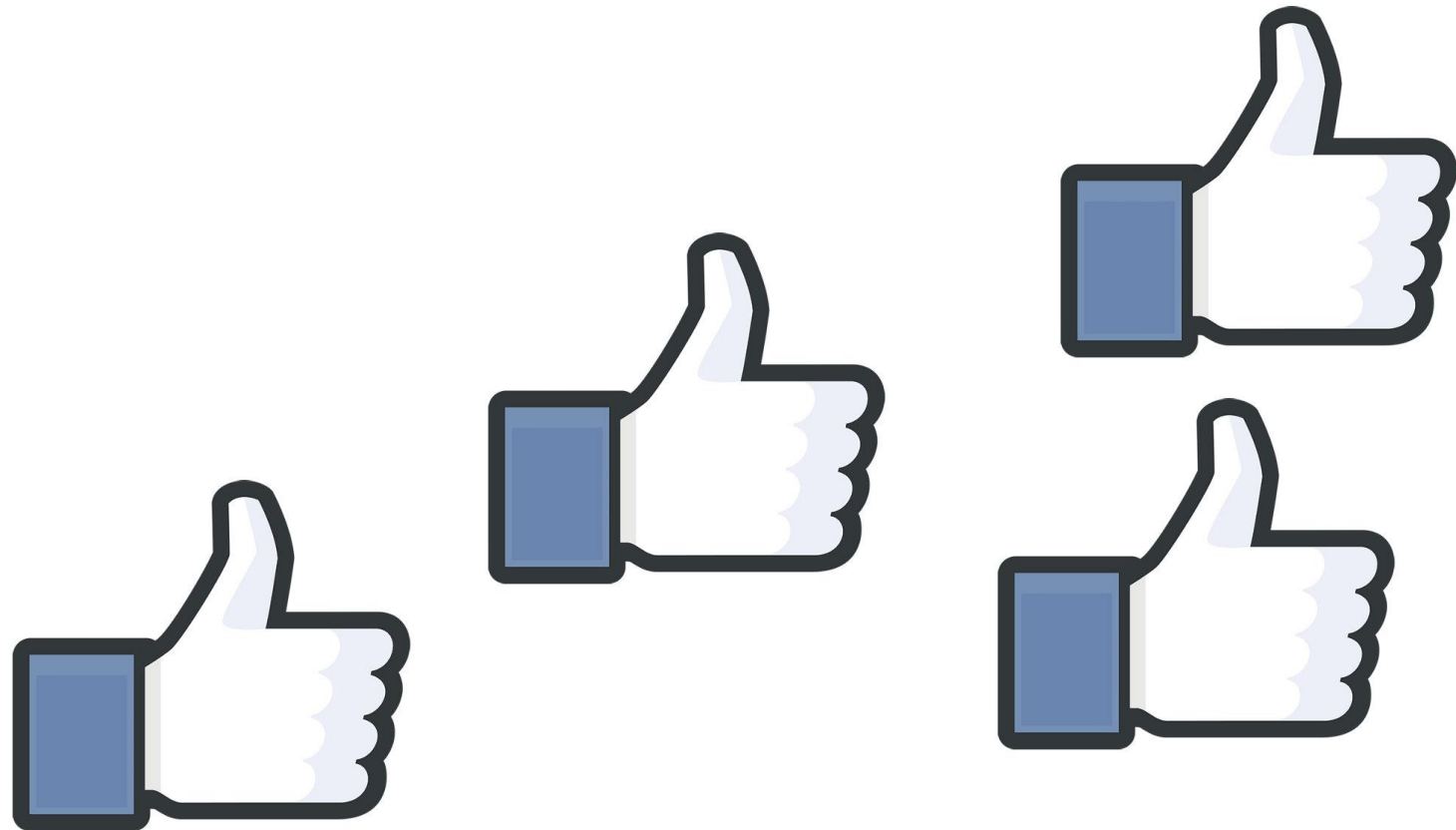
- Nome é **único** e serve para invocar a função
- Pode ser **invocada** a partir de outras funções
- Deve realizar uma **única tarefa** bem definida
- Comporta-se como uma **caixa preta**
- Deve ser o mais **independente** possível do restante do programa
- Pode receber **parâmetros** de quem a invocou
- Pode **retornar**, para quem a invocou, um valor como resultado do seu funcionamento
- Um programa em C tem **sempre** que possuir a função **main**

# Funções – Como funcionam

- O código da função só é executado quando esta é **invocada** em alguma parte do programa
- Sempre que uma função é invocada, o programa que a invoca é temporariamente suspenso, voltando a executar após o término da função
- O programa que invoca uma função pode enviar **argumentos**, que são recebidos por variáveis locais chamadas **parâmetros**
- Ao terminar, uma função pode **devolver** um valor para quem a invocou

# Funções – Vantagens

- Facilitam a leitura, escrita, compreensão, modificação, manutenção e correção de programas



# Funções - Exercício

- Modifique a função linha para imprimir uma determinada quantidade de asteriscos





# Funções - Exercício


- Modifique a função linha para imprimir uma determinada quantidade de asteriscos
- Função: **linha**
- Quantos parâmetros vai receber? **1**
- Qual o tipo do parâmetro? **Inteiro**
- Variável para armazenar o parâmetro? **Num**
- Cabeçalho da função?
- Corpo da função?



# Parâmetros

- A comunicação com uma função se faz por meio dos argumentos que lhe são enviados e dos parâmetros presentes na função que os recebe
- Parâmetro é uma variável local pertencente à função
  - Iniciada com o valor enviado pelo programa que invoca a função

```
main()  
{  
    ...  
    funcao('A' , 123 , 23.456);  
    ...  
}  
funcao(char ch, int n, float x)  
{  
    ...  
}
```



# Parâmetros

- O nome dos parâmetros é totalmente independente do nome das variáveis que lhe são passadas como argumento
- O cabeçalho de uma função NUNCA deve ser seguido por ponto e vírgula
- Não se pode definir funções dentro de funções em C

# Funções - Exercício

- Modifique a função linha para imprimir uma determinada quantidade de símbolos informados pelo usuário



# Corpo da Função

- O corpo da função é um conjunto de instruções em C escrito em um bloco após o cabeçalho
- A função é executada até que o corpo termine ou até que a instrução `return` seja encontrada

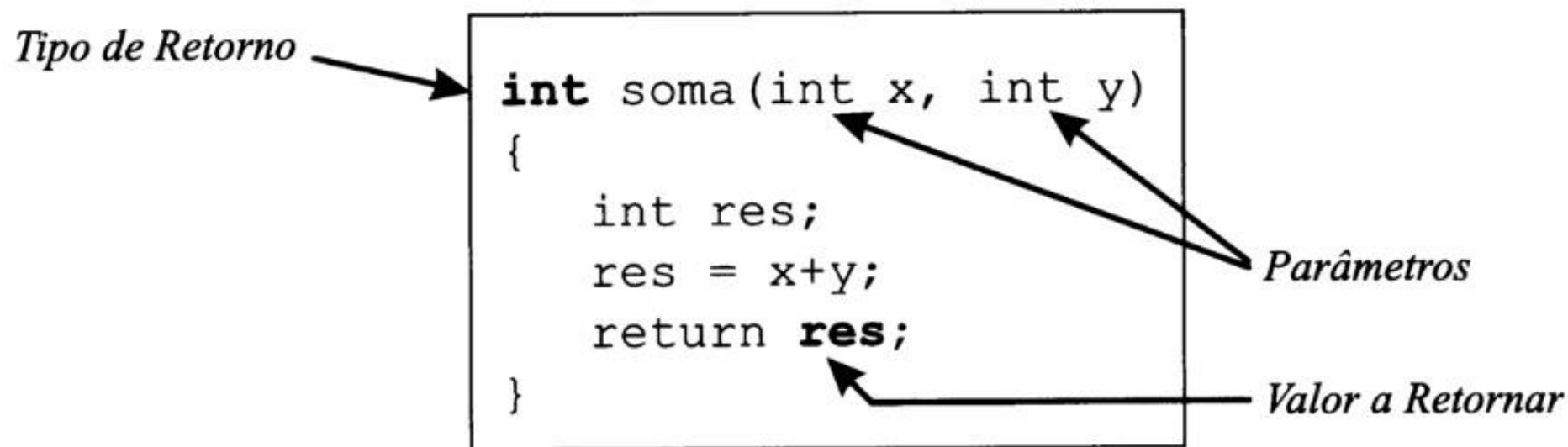
# Exercício

- Escreva um programa em C que implemente as seguintes funções:
- `abs(x)`: Retorna o módulo de  $x$
- `impar(x)`: retorna verdadeiro se  $x$  é ímpar
- `fatorial(x)`: retorna o valor de  $x!$
- `Imprime(x)`: imprime  $x$  na tela
- `imprimeFatorial(x)`: imprime na tela  $x!$
- `Combinacao(x,y)`: retorna  $x$  combinado  $y$  a  $y$



# Funções que Retornam um Valor

- Algumas funções podem ser definidas de modo que devolvam UM ÚNICO resultado
  - Resultado pode ser armazenado em uma variável ou aproveitado por qualquer instrução



# Funções que Retornam um Valor

- Quando o tipo de retorno da função não é declarado, assume-se que é int
- Tipos de retorno:
  - char
  - int
  - float
  - double
  - void (procedimentos)



# Variáveis Locais

- Variáveis podem ser declaradas dentro do corpo de uma função
- Variáveis locais, isto é, conhecidas apenas no escopo da função
  - Destruídas após a execução da função

```
função( ..... )  
{  
    declaração de variáveis  
    instruções  
}
```

# Variáveis Globais

- Acessíveis por todas as funções do programa
- Declaração é feita fora do escopo das funções
- Cuidado com efeitos colaterais!
  - Função altera o valor de uma variável global

# Exercício

- Escreva um programa em C que construa o triângulo de Pascal com n linhas, em que n é informado pelo usuário

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
```

