

# MCTA028-15: Programação Estruturada

# Aula 5: Recursão (Segunda Parte)

Wagner Tanaka Botelho wagner.tanaka@ufabc.edu.br / wagtanaka@gmail.com Universidade Federal do ABC (UFABC) Centro de Matemática, Computação e Cognição (CMCC)

# Introdução

## Introdução

- Na Linguagem C:
  - Uma função pode chamar outras funções;
  - O main() pode chamar qualquer função, como por exemplo o printf() ou uma função definida pelo programador:
    - A função implementada pelo programador também pode chamar outras funções.
- Vocês sabiam que uma função pode, inclusive, chamar a si própria?
  - Isso é conhecido como RECURSIVIDADE, que é o processo de REPETIR alguma coisa de maneira SIMILAR.

```
Ex 04.c X
      #include<stdio.h>
 1
     dint Soma(int a, int b) {
          int result=0;
          result = a + b;
          return result;
10
     □void main(){
          int x=0, y=0, result=0;
13
          printf("Digite dois numeros: \n");
14
15
          scanf("%i", &x);
16
          scanf("%i", &y);
17
18
          result = Soma(x, y);
19
          printf("O resultado da soma eh: %i", result);
20
21
```

## Exemplo no Mundo Real

+ Para melhor compreender o conceito, vamos utilizar um exemplo no MUNDO REAL:





2





**Etapa 1:** O vaso está vazio?



Etapa 2:

Não! Então, tira-se uma flor!



3





**Etapa 1:** O vaso está vazio?



Etapa 2:

Não! Então, tira-se uma flor!



3

O problema é: Como esvaziar um vaso contendo NENHUMA flor?

Etapa 1: O vaso está vazio?



Etapa 2:

Sim! Então, FIM do processo!



Percebe-se que TODO o processo de REMOVER flores é MUITO REPETITIVO!!

Para esvaziar um vaso contendo N flores, primeiro verifica-se se o vaso está vazio. Se o vaso não estiver vazio, tira-se uma flor (N-1).

```
*Ex 05.c 🗶
      #include<stdio.h>
       void esvaziar(int gtdFlores)
           if (atdFlores>0
               atdFlores-
               printf("Otd. de flores no vaso: %i\n", qtdFlores);
               esvaziar(gtdFlores);
10
11
     ∃void main(){
12
           int qtd=3;
13
14
          printf("Otd. de flores no vaso: %i\n", qtd);
15
          esvaziar(qtd);
16
```

Linha	qtd	qtdFlores
12	3	
14	{3}	
3		3
5		<b>3 2</b>
6		{2}
3		1
5		1
6		<b>{1}</b>
3		1
5		<b>8</b>
6		{0}
3		0

## **Exemplo do Fatorial**

#### Como calcular o fatorial de 3?



Para calcular o fatorial de 3, multiplica-se o número 3 pelo fatorial de 2 (definido como 2!).



Generalizando esse processo, tem-se que o fatorial de N é igual a N multiplicado pelo fatorial de (N - 1), ou seja, N! = N \* (N - 1)!.



Quando o processo termina?

Quando o número zero for atingido. Nesse caso, o valor do fatorial de 0 (0!) é definido como igual a 1.



O fatorial de 3 é o produto de todos os números inteiros entre 1 e 3.



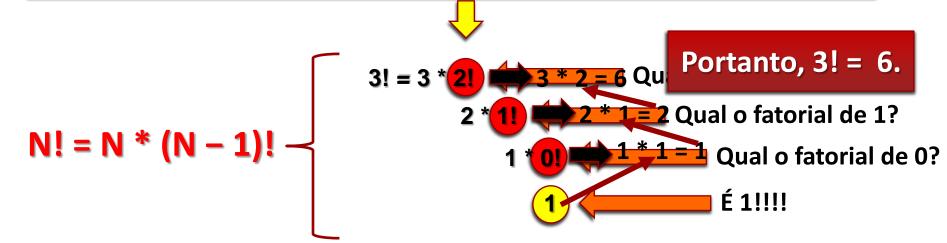
Aplicando a ideia da RECURSÃO!!

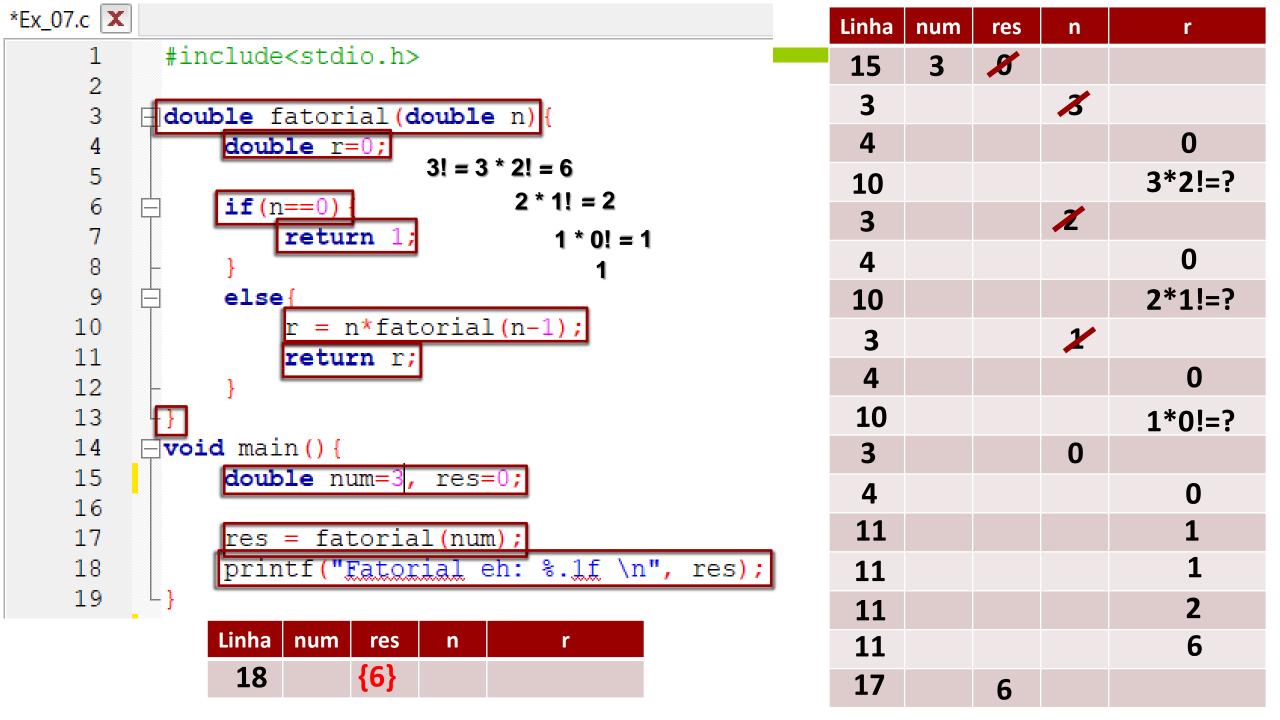
**Etapa 1:** O fatorial de 3 é definido em função do fatorial de 2;

**Etapa 2:** O fatorial de 2 é definido em função do fatorial de 1;

Etapa 3: O fatorial de 1 é definido em função do fatorial de 0;

Etapa 4: O fatorial de 0 é definido como igual a 1.





### Recursão

### Recursão

- As formas recursivas são consideradas "mais enxutas" e "mais elegantes" do que as formas interativas;
- Muito cuidado na implementação de funções recursivas, pois DOIS critérios devem ficar bem estabelecidos:
  - Critério de parada;
  - Parâmetro da chamada recursiva.
- Critério de parada:
  - Determina quando a função deve PARAR de CHAMAR a si mesma. Se ela não existir, a função continuará executando até esgotar a memória do computador (*loop* infinito).;
  - No cálculo do fatorial, o critério de parada ocorre quando tenta-se calcular o fatorial de zero: 0!=1.

### Recursão

- Parâmetro da chamada recursiva:
  - O valor do parâmetro passado deve SEMPRE ser mudado, para que a recursão chegue a um término;
  - Se o parâmetro for sempre o mesmo, a função continuara executando até esgotar a memória do computador (*loop* infinito);
  - No cálculo do fatorial, a mudança no parâmetro da chamada recursiva ocorre quando o fatorial de N é definido em termos do fatorial de (N-1): N! = N\*(N-1)!.

17/18

```
*Ex_07.c X
       #include<stdio.h>
       double fatorial(double n) {
 4
           double r;
           if(n==0){
                                    Critério de parada!
                return 1;
           else{
                                               Parâmetro do fatorial
                r = n*fatorial(n-1);
10
                                               sempre muda!
11
                return r;
12
13
       void main() {
14
15
           double num=3, res=0;
16
           res = fatorial(num);
17
           printf("Fatorial eh: %.1f \n", res);
18
19
```

### Referências

- SALES, André Barros de; AMVAME-NZE, Georges. Linguagem C: roteiro de experimentos para aulas práticas. 2016;
- BACKES, André. Linguagem C Completa e Descomplicada. Editora Campus. 2013;
- SCHILDT, Herbert. C Completo e Total. Makron Books. 1996;
- DAMAS, Luís. Linguagem C. LTC Editora. 1999;
- DEITEL, Paul e DEITEL, Harvey. C Como Programar. Pearson. 2011.