Recursão

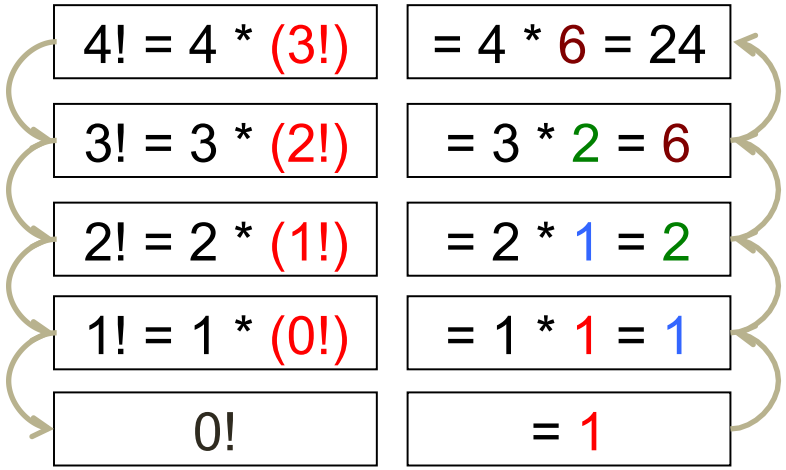
Departamento de Eletrônica e Computação
Dr. Osmar Marchi dos Santos

Conceitos básicos de recursão

- Permite definir um problema em termos de uma ou mais versões ***iguais ou menores*** do mesmo problema
- Pode ser aplicada sempre que o problema sendo resolvido pode ser definido em função de si próprio
- Exemplo clássico ... Fatorial!

Problema Fatorial

- Cálculo do fatorial de um número segue:
- $n! \begin{cases} 1, & n = 0 \\ n * (n - 1)!, & n > 0 \end{cases}$
- Por exemplo, para 4! temos:



Fatorial sem recursividade

```
int fatorial(int n) {
  int i, fat;
  fat = 1;
  if (n > 0) {
    for (i = 1; i < = n; i++) {
      fat = fat * i;
    }
  }
  return fat;
}
```

Fatorial com recursividade

```
int fatorial(int n) {
  if (n > 0) {
    return (n * fatorial(n – 1));
  }
  return 1;
}
```

Fatorial com recursividade

fatorial(4)

=> (4 > 0) RETORNE 4 * fatorial(3)

=> (3 > 0) RETORNE 3 * fatorial(2)

=> (2 > 0) RETORNE 2* fatorial(1)

=> (1 > 0) RETORNE 1* fatorial(0)

=> (0 = 0)

<= RETORNE 1

<= RETORNE 1 * 1 // 1

<= RETORNE 2 * 1 // 2

<= RETORNE 3 * 2 // 6

<= RETORNE 4 * 6 // 24

Cuidados com a recursão

- Em funções recursivas pode ocorrer um erro de não terminação do algoritmo, como um laço *(loop)* infinito

- Para garantir a terminação das repetições, deve-se:
  - Definir uma função que implica em uma condição de terminação
  - Garantir que a função decresce a cada passo de repetição, permitindo que a condição de terminação seja atingida

- Qual a condição de terminação do cálculo de fatorial?

Estrutura de uma recursão

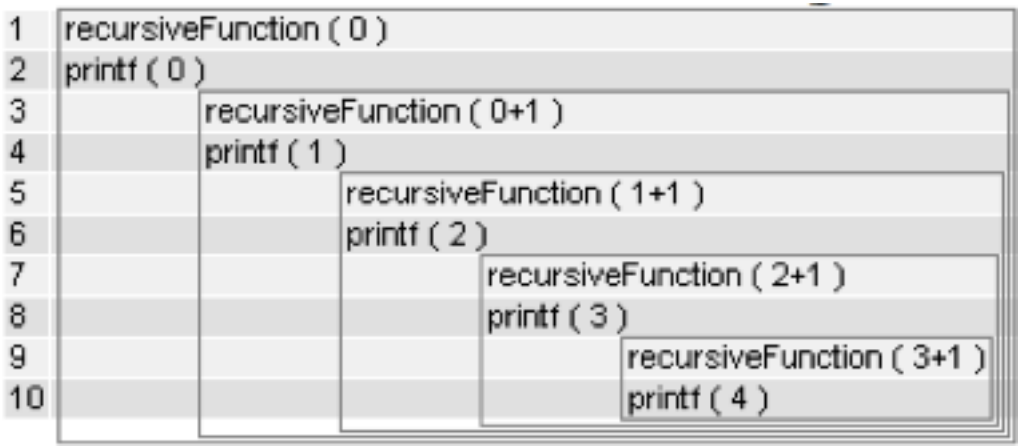
- Uma recursão obedece uma estrutura que deve conter os seguintes elementos (não necessariamente na mesma ordem!):

- return função(parâmetro(s))
  - Teste de término de recursão utilizando parâmetro
    - Se teste OK, retorna término
  - Processamento
    - Processa as informações do parâmetro
  - Chamada recursiva utilizando parâmetro
    - **Parâmetro deve ser modificado, de forma que a recursão chegue a um término**

- Foi utilizada essa estrutura para resolver o problema fatorial?

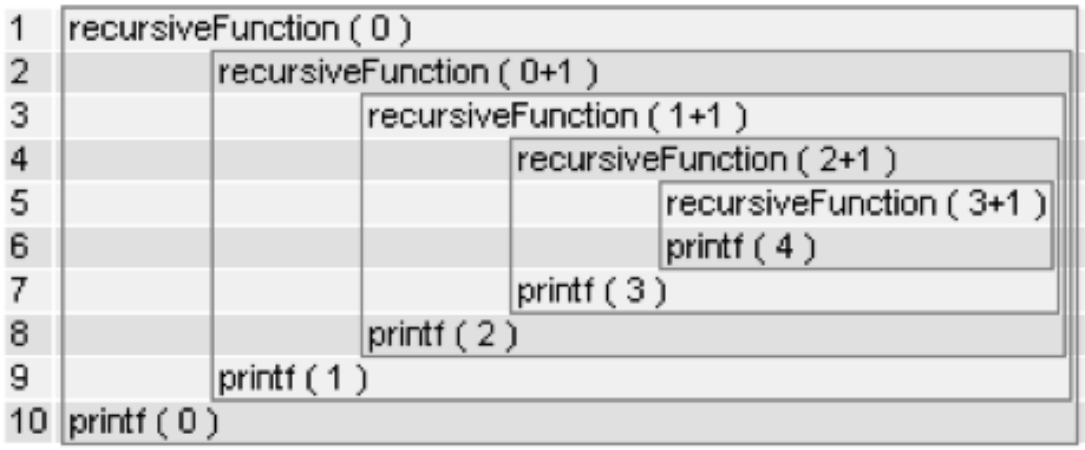
Ordem de chamada de funções

```
void recursiveFunction(int num) {
  if (num < 5) {
    printf("num = %d", num);
    recursiveFunction(num+1);
  }
}
```



Ordem de chamada de funções

```
void recursiveFunction(int num) {
  if (num < 5) {
    recursiveFunction(num+1);
    printf("num = %d", num);
  }
}
```



Vantagens da recursão

- Vantagens ☺
  - A maior parte dos problemas pode ser resolvido com ou sem recursão. Porém, em algumas situações é obrigatório (ou facilita muito o desenvolvimento) o uso de recursão. Por exemplo, algoritmos que trabalham com árvores e grafos
  - Ao utilizar recursão, o tamanho do programa é reduzido, criando uma solução mais compacta (elegante) que a versão iterativa

Desvantagens da recursão

- Mas como tudo na vida, desvantagens ☹
  - Requer espaço extra de armazenamento. As chamadas recursivas são alocadas na pilha. Para cada chamada, uma memória separada é alocada (as variáveis das funções são específicas, locais a cada função)
  - Se esquecer de incluir uma condição de parada na função, o algoritmo vai executar fora de memória (*CTRL +ALT+DEL, Segmentation fault, etc ...*)
  - A função de recursão não é eficiente em termos de velocidade e tempo

- Por que não é eficiente em termos de velocidade e tempo?

Exercícios – Usando recursão

- Criar um algoritmo de contagem regressiva! Dado um parâmetro inteiro, ele vai decrementado até chegar no valor 0.
- Realizar a soma dos 100 primeiros números inteiros.
- Determinar o valor do elemento máximo de um vetor int v[100].
- Fazer a soma de todos os valores de um vetor int v[100].
- Descreva uma função para calcular exponenciação (base 2). Exemplo: $2^3 = 2 * 2 * 2 = 8$

Exercícios – Usando recursão

- Criar uma função para calcular a série de Fibonacci de um determinado número passado como parâmetro.

- Crie uma função para calcular o MDC (Máximo Divisor Comum) de dois números passados por parâmetro:

- mdc (x, y) =

y	se x>= y e (x resto y) = 0
mdc(y,x)	se x < y
mdc(y, (x resto y))	caso contrário

- Exemplo: mdc(9,6) retorna 3.