Departamento
de Ciência da
COMPUTAÇÃO

UDESC
Joinville

Linguagem C
Funções
(recursividade)

André Tavares da Silva

andre.silva@udesc.br

Recursividade

- Na linguagem C, assim como na maioria linguagens de programação atuais, uma função pode chamar a si mesma. Uma função assim é chamada função **recursiva**.
- Existem muitos problemas que são definidos com base neles mesmos, ou seja, podem ser descritos por instâncias do próprio problema. Para estas classes de problemas, o conceito de recursividade torna-se muito útil. Ex.: n! = n * (n-1)!

Recursividade

- Deve-se tomar alguns cuidados ao se fazer funções recursivas.
- Um cuidado importante a se verificar, é a existência de um **critério de parada** para determinar quando a função deverá parar de chamar a si mesma. Este cuidado serve para impedir que a função se chame infinitas vezes.

Recursividade

- No caso da função fatorial, podemos escrevê-la da seguinte forma:

```
int fatorial(int n) {
    return n * fatorial(n-1);
}
```

- Qual o critério de parada?

Recursividade

- No caso da função fatorial, podemos escrevê-la da seguinte forma:

```
int fatorial(int n) {
    if(n<=1)
        return 1;
    return n * fatorial(n-1);
}
```

Exercícios

- Crie um programa em C que peça um número inteiro ao usuário e retorne a soma de todos os números de 1 até o valor informado pelo usuário, ou seja: 1+2+3+...+n. Utilize recursividade.
- Faça uma função que calcule o n-ésimo termo da sequência de Fibonacci.

  $f(1) = 1$

  $f(2) = 1$

  $f(n) = f(n-1) + f(n-2)$
- Refaça a função anterior usando recursividade.

Exercícios

- Na linguagem C, existe a função *pow* que recebe dois valores e realiza o cálculo de potenciação (exponenciação), ou seja, eleva o primeiro valor à potencia indicada pelo segundo e retorna a resposta calculada. Implemente uma função recursiva para cálculo de exponenciação utilizando apenas operadores aritméticos de soma, subtração, multiplicação e divisão.