

# INF1005: Programação 1

## Funções Recursivas



# Tópicos Principais

- Recursão
  - Definições recursivas
- Funções Recursivas
  - Implementação
  - Comportamento

# Definições Recursivas

- Em uma definição recursiva um item é definido em termos de si mesmo, ou seja, o **item que está sendo definido aparece como parte da definição**;
- Em todas as funções recursivas existe:
  - Caso base (um ou mais) cujo resultado é imediatamente conhecido.
  - Passo recursivo em que se tenta resolver um sub-problema do problema inicial.

# Definições Recursivas

- **Exemplo:** o fatorial de um número

$$n! = \begin{cases} 1, & \text{se } n = 0 \\ n \times (n - 1)!, & \text{se } n > 0 \end{cases}$$

Caso BASE



Passo  
Recursivo

# Definições Recursivas

- **Exercício:** forneça a definição recursiva para a operação de potenciação

$$x^n = \begin{cases} 1, & \text{se } n = 0 \\ x \times x^{(n-1)}, & \text{se } n > 0 \end{cases}$$

Caso BASE



Passo  
Recursivo

# Funções Recursivas

- Definição:
  - Uma função recursiva é aquela que faz uma chamada para si mesma. Essa chamada pode ser:
    - direta: uma função **A** chama a ela própria
    - indireta: função **A** chama uma função **B** que, por sua vez, chama **A**

```
/* Recursao direta */  
void func_rec(int n)  
{  
    ...  
    func_rec(n-1);  
    ...  
}
```

# Funções Recursivas

- Exemplo: função recursiva para cálculo de fatorial


$$n! = \begin{cases} 1, & \text{se } n = 0 \\ n \times (n-1)!, & \text{se } n > 0 \end{cases}$$

```
/* Função recursiva para cálculo do fatorial */  
int fat (int n)  
{  
    if (n==0)  
        return 1;  
    else  
        return n*fat(n-1);  
}
```

Caso BASE



Passo  
Recursivo



# Funções Recursivas

- Exemplo: função recursiva para cálculo de potenciação

$$x^n = \begin{cases} 1, & \text{se } n = 0 \\ x \times x^{(n-1)}, & \text{se } n > 0 \end{cases}$$

```
/* Função recursiva para cálculo de potenciacao */  
int pot (int x, int n)  
{  
    if (n==0)  
        return 1;  
    else  
        return x*pot(x,n-1);  
}
```

Caso BASE

Passo Recursivo



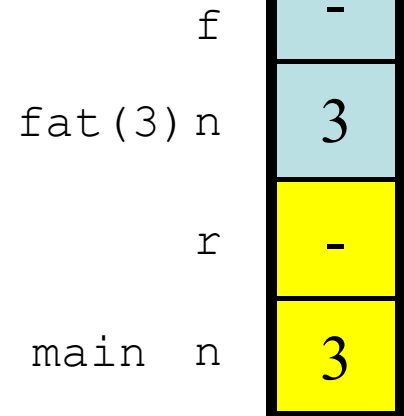
# Funções Recursivas

- Comportamento:
  - quando uma função é chamada recursivamente, cria-se um ambiente local para cada chamada
  - as variáveis locais de chamadas recursivas são independentes entre si, como se estivéssemos chamando funções diferentes

# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

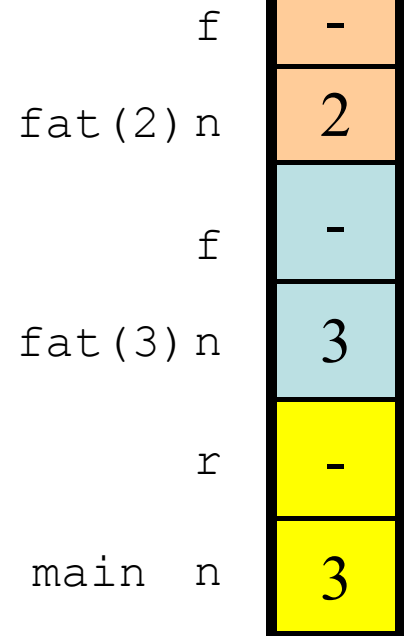
/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```



# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```



# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

f

-

fat(1) n

1

f

-

fat(2) n

2

f

-

fat(3) n

3

r

-

main n

3

# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

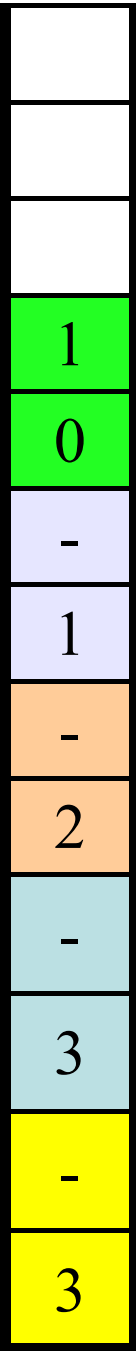
f	-	
fat(0) n	0	
f	-	
fat(1) n	1	
f	-	
fat(2) n	2	
f	-	
fat(3) n	3	
r	-	
main n	3	

# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

f  
fat(0) n  
f  
fat(1) n  
f  
fat(2) n  
f  
fat(3) n  
r  
main n

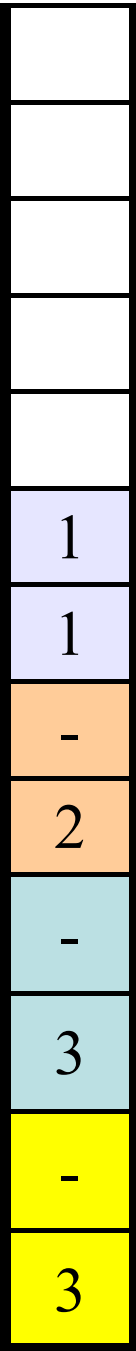


# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

f  
fat(1) n  
f  
fat(2) n  
f  
fat(3) n  
r  
main n

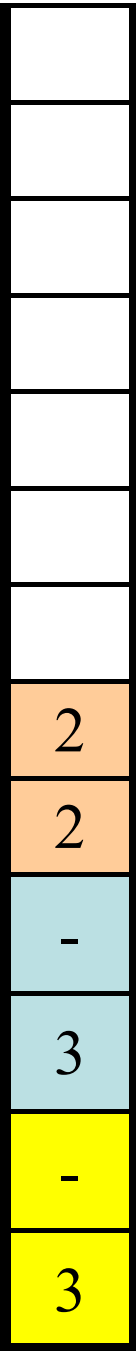


# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

f  
fat(2) n  
f  
fat(3) n  
r  
main n



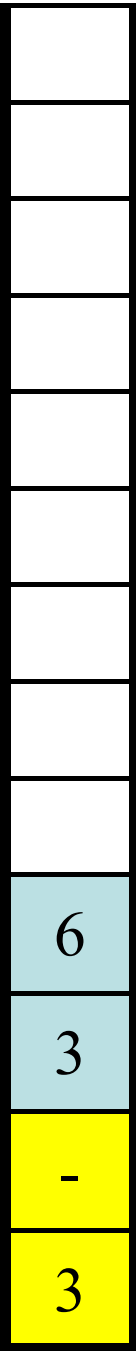


# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```

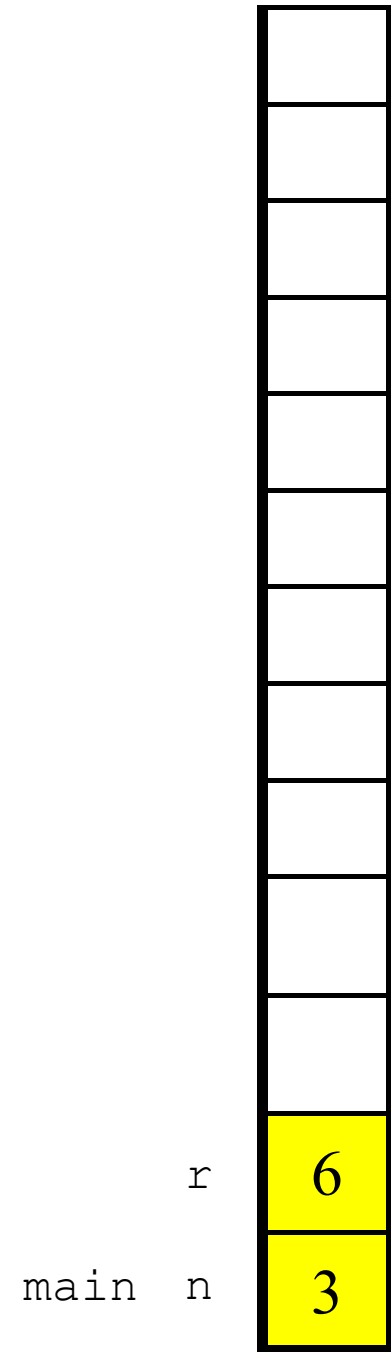
f  
fat(3) n  
r  
main n



# Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{   int n = 3;
    int r;
    r = fat ( n );
    printf("Fatorial de %d = %d \n", n, r);
    return 0;
}

/* Função recursiva para cálculo do fatorial */
int fat (int n)
{
    int f;
    if (n==0)
        f=1;
    else
        f= n*fat(n-1);
    return f;
}
```



# Funções Recursivas

- Exemplo: série de Fibonacci

$$fib(n) = \begin{cases} 0, & \text{se } n = 0 \\ 1, & \text{se } n = 1 \\ fib(n-1) + fib(n-2), & \text{se } n > 1 \end{cases}$$

Diagram illustrating the recursive definition of the Fibonacci function:

- Two orange boxes labeled "2 casos BASE" point to the base cases:  $0, \text{ se } n = 0$  and  $1, \text{ se } n = 1$ .
- An orange box labeled "Passo Recursivo" points to the recursive case:  $fib(n-1) + fib(n-2), \text{ se } n > 1$ .

# Funções Recursivas

- Exemplo: série de Fibonacci

```
/* Calculo da serie de Fibonacci */  
int fib (int n)  
{  
    if (n==0)  
        return 0;  
    else if (n==1)  
        return 1;  
    else  
        return (fib(n-1) + fib(n-2));  
}
```

# Referências

Waldemar Celes, Renato Cerqueira, José Lucas Rangel,  
*Introdução a Estruturas de Dados*, Editora Campus  
(2004)

- Capítulo 4 – Funções