Programação de Computadores II

Recursividade

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

Slides adaptados dos originais dos profs.: Marco Antonio Casanova e Marcelo Gattass (PUC-Rio)

Referências

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

O assunto de recursividade é abordado nos capítulos:

Capítulo 4 – Funções

Capítulo 7 – Cadeias de Caracteres

Capítulo 16 – Ordenação

Capítulo 17 – Busca

Funções Recursivas

- Tipos de recursão:
  - direta:
    - uma função A chama a ela própria
  - indireta:
    - uma função A chama uma função B que, por sua vez, chama A
- 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

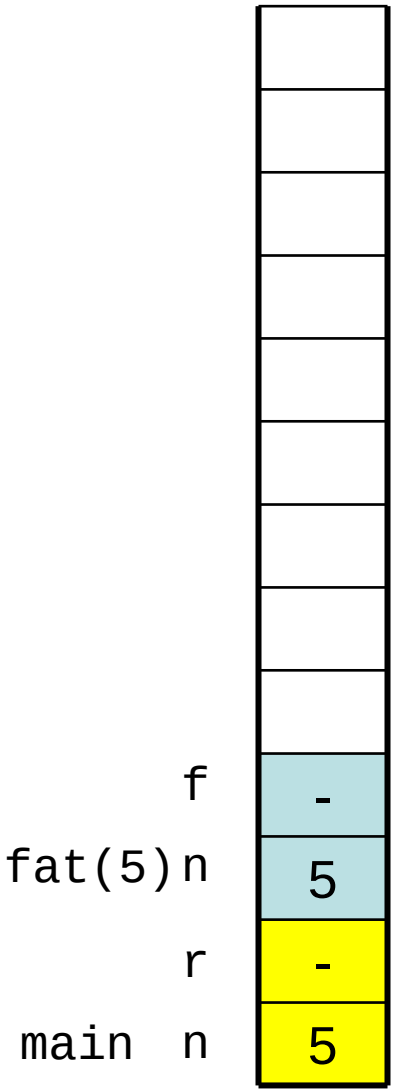
- Exemplo: definição recursiva de fatorial

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

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

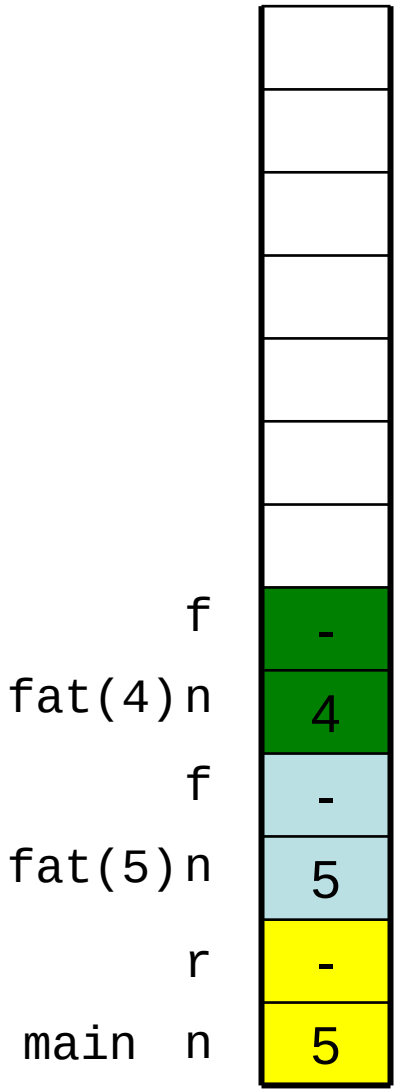
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



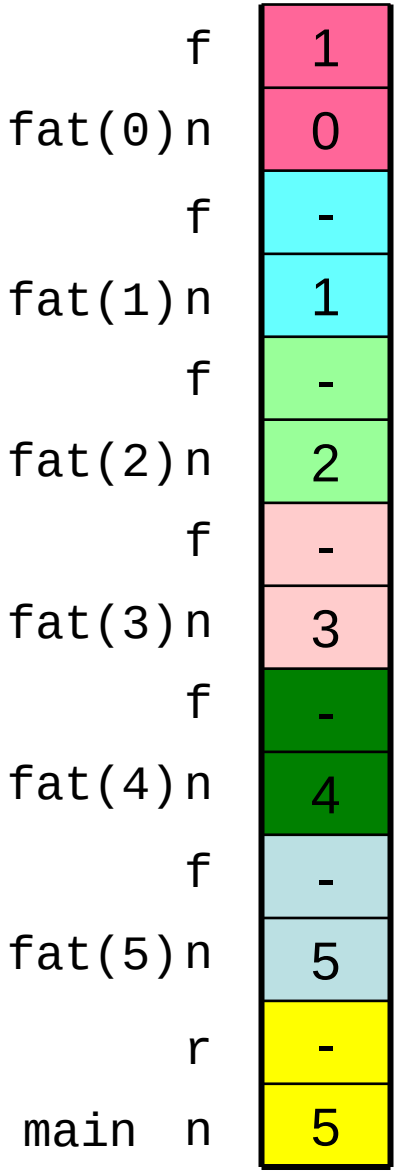
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



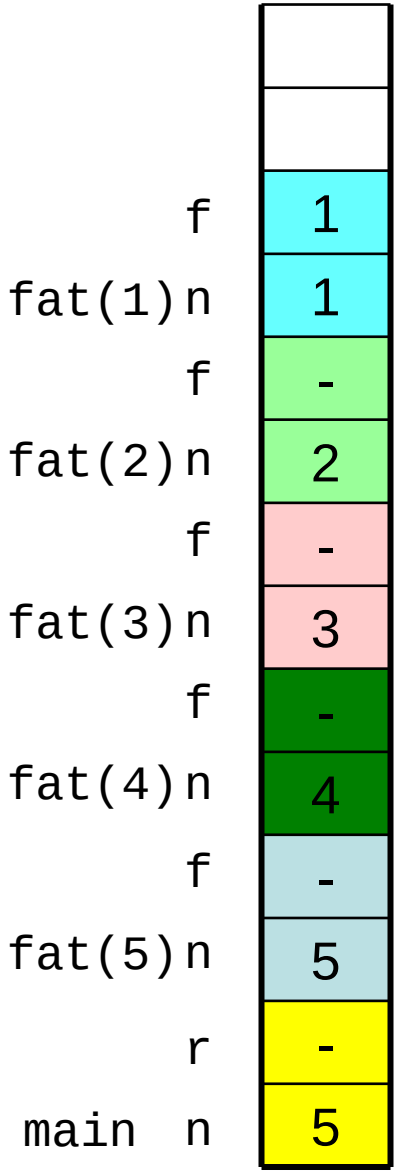
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



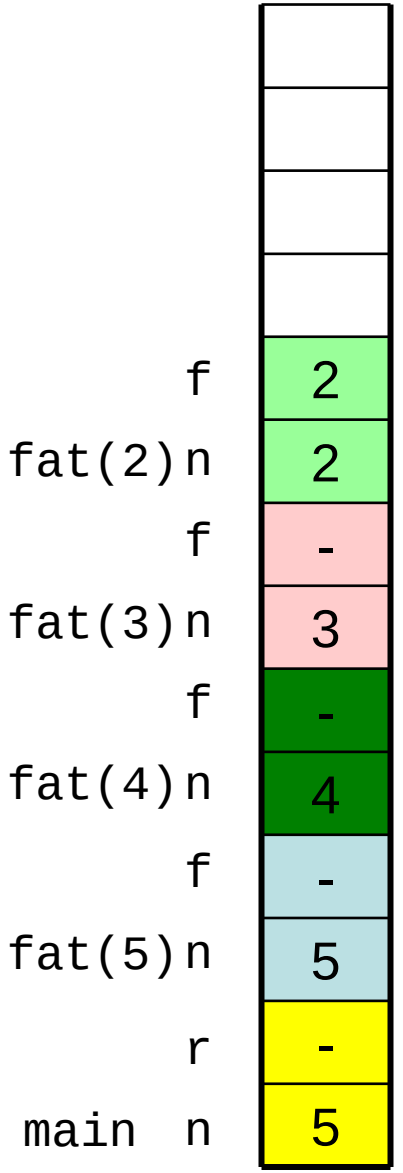
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



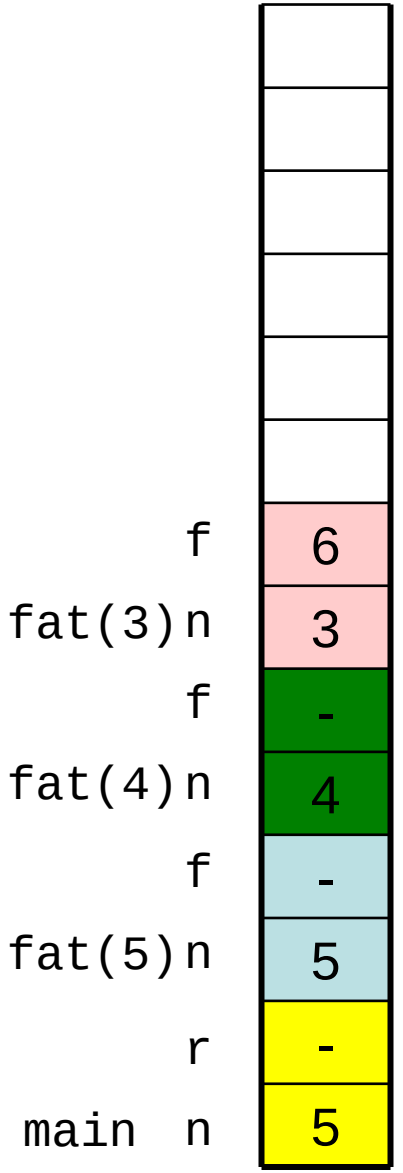
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



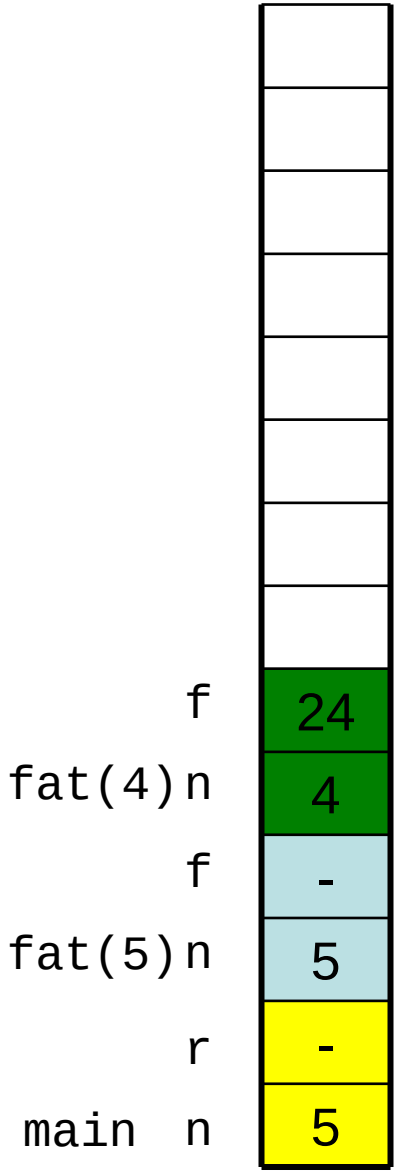
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



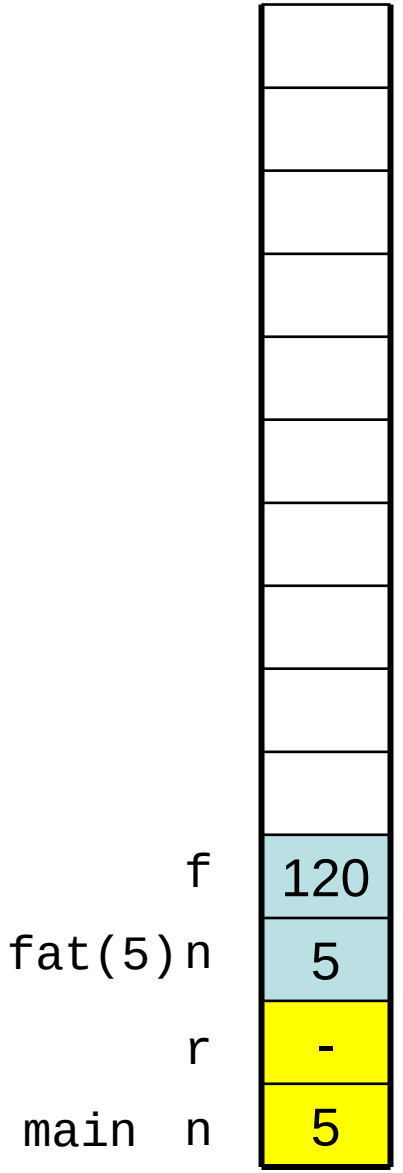
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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;
}
```



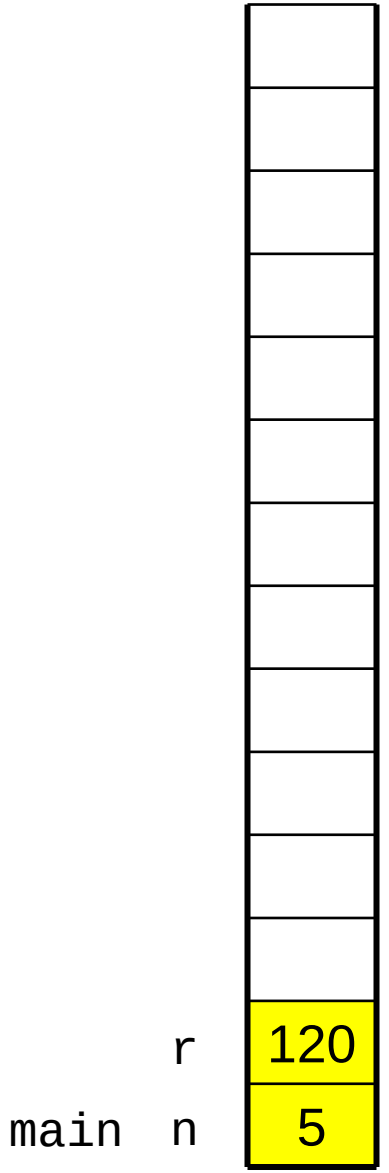
Funções Recursivas

```
#include <stdio.h>
int fat (int n);
int main (void)
{  int n = 5;
   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 = 5;
   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;
}
```



Cadeias de caracteres

- Funções recursivas para manipular cadeias de caracteres:
  - baseiam-se em uma definição recursiva de cadeias de caracteres:

    Uma cadeia de caracteres é:

    - *a cadeia de caracteres vazia; ou*
    - *um caractere seguido de uma cadeia de caracteres*

Cadeias de caracteres

- Implementação recursiva da função “imprime” e imprime invertido:

```
void imprime_rec (char* s) {
   if (s[0] != '\0') {
      printf(“%c”, s[0]);
      imprime_rec(&s[1]);
   }
}
```

```
void imprime_inv (char* s) {
   if (s[0] != '\0') {
      imprime_inv(&s[1]);
      printf(“%c”, s[0]);
   }
}
```

Cadeias de caracteres

- Implementação recursiva da função “comprimento”:

```
int comprimento_rec (char* s)
{
   if (s[0] == '\0')
      return 0;
   else
      return 1 + comprimento_rec(&s[1]);
}
```

Exemplo

```
int mdc_recursiva(int a, int b) {
    printf("\n%d e %d", a, b);
    if (a % b == 0) return b;
    return mdc_recursiva (b, a % b);
}
int main() {
    int a=32, b=18;
    printf("\nResultado=%d",mdc_recursiva(a, b));
}
```

```
32 e 18
18 e 14
14 e 4
4 e 2
Resultado=2
```

Busca Binária em Vetor Recursiva

- dois casos a tratar:
  - busca deve continuar na primeira metade do vetor:
    - chamada recursiva com parâmetros:
      - o número de elementos da primeira parte restante
      - o mesmo ponteiro para o primeiro elemento (pois a primeira parte tem o mesmo primeiro elemento do que o vetor como um todo)
  - busca deve continuar apenas na segunda parte do vetor:
    - chamada recursiva com parâmetros:
      - número de elementos restantes
      - ponteiro para o primeiro elemento dessa segunda parte
    - valor retornado deve ser corrigido

```
int busca_bin_rec (int n, int* vet, int elem)
{
   /* testa condição de contorno: parte com tamanho zero */
   if (n <= 0)
      return -1;
   else {
      int meio = n/2;
      if (elem < vet[meio])
         return busca_bin_rec(meio,vet,elem);
      else if (elem > vet[meio]) {
         int r = busca_bin_rec(n-1-meio, &vet[meio+1],elem);
         if (r==-1)
            return -1;
         else
            return (meio+1+r);  /* correção da origem */
      }
      else  /* elem==vet[meio] */
         return meio;      /* elemento encontrado */
   }
}
```