MAUÁ ESCOLA DE ENGENHARIA

11

Recursividade
Quicksort

Exemplo de Recursividade

Do dicionário

Recursividade:

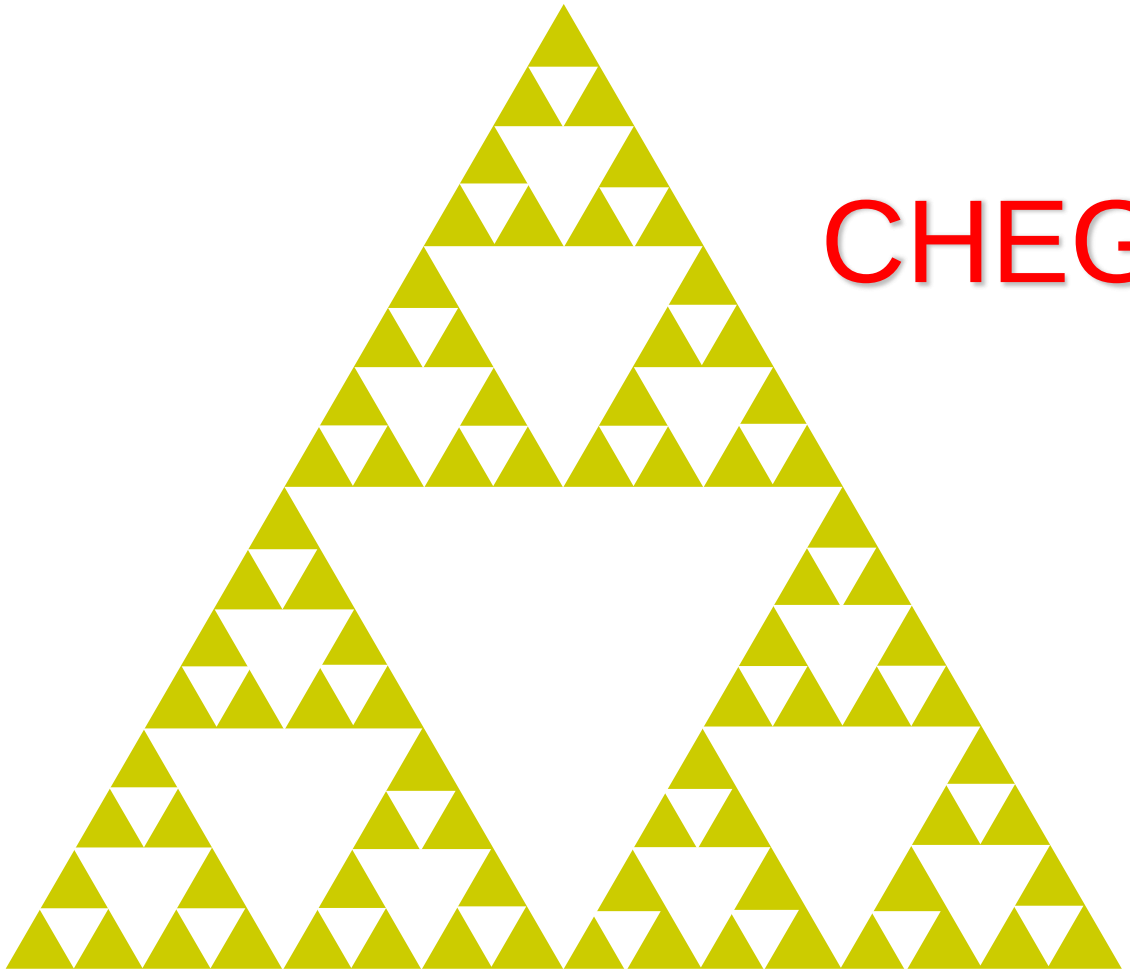
Veja Recursividade. Se não entender, veja Recursão.

Recursão:

Veja Recursividade.

Exemplo de Recursividade

Fractal – Triângulo de Sierpinski



Recursividade Matemática

Um objeto é dito recursivo quando é definido em termos de um caso mais simples de si mesmo;

Muitas funções de interesse são definidas desta forma.

Exemplos:

Fatorial de um número natural ($n \in N$)

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

Exemplos:

Multiplicação de números naturais (a, b ∈ N)

$$m(a,b) = \begin{cases} 0 & , \text{se } b = 0 \\ a + m(a, b-1), & \text{se } b \neq 0 \end{cases}$$

Número de Fibonacci (n ∈ N)

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

Recursividade Matemática

Exercite:

f(6) =

m(5,3) =

F(9) =

Sub-rotinas Recursivas

Assim como na Matemática, na linguagem C uma função é classificada como recursivos quando é definida em termos de um caso mais simples de si mesmo.

Exemplos

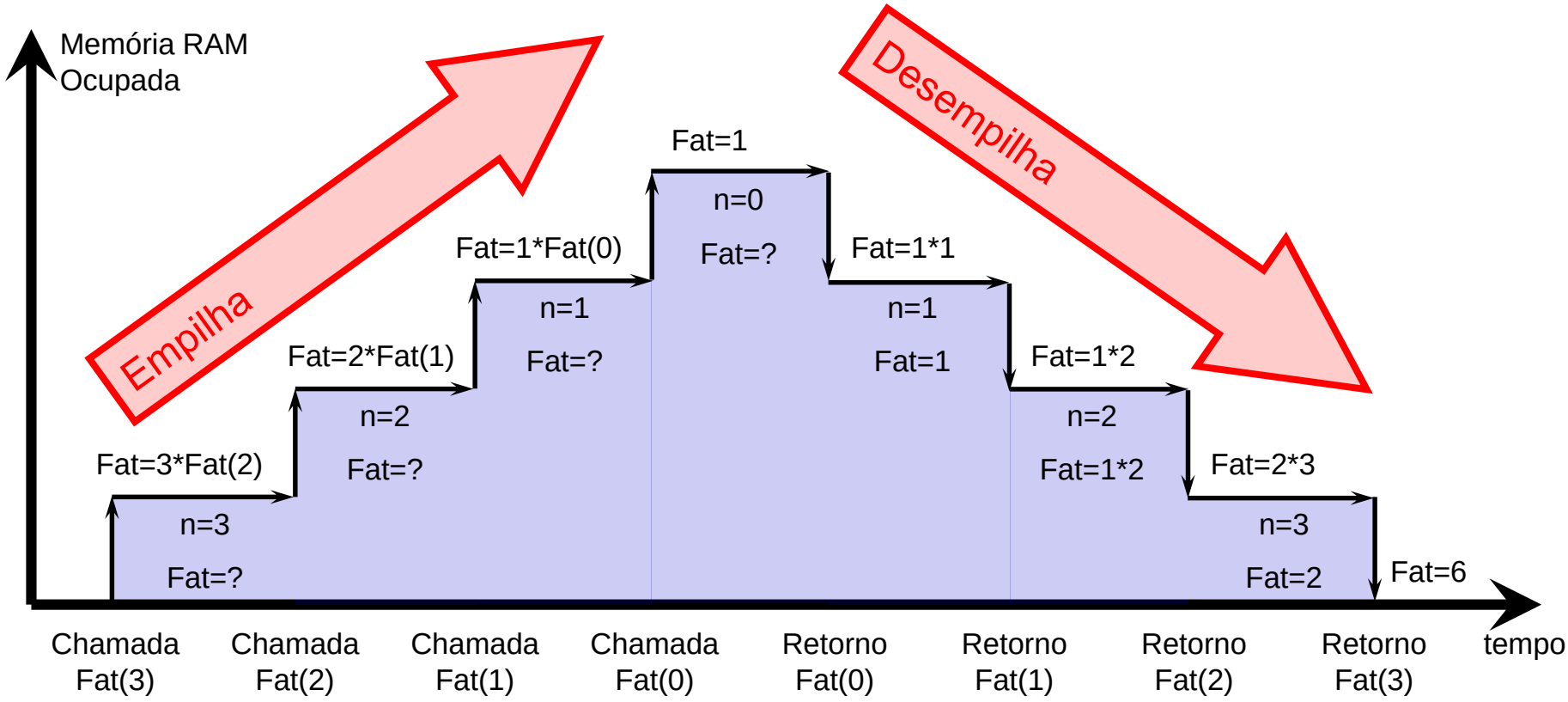
Fatorial de um número natural ($n \in N$)

```
int fatorial(int n){
    if (n == 0)
        return 1;
    else
        return n*fatorial(n-1);
}
```

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

Alocação de Memória

Veja como é feita a alocação de memória em um algoritmo recursivo.



Vantagens e Desvantagens

- Grande facilidade na implementação de algoritmos com características recursivas;
- grande consumo de memória;
- normalmente, o tempo de execução é maior do que em algoritmos iterativos.

Fibonacci(50)	Iterativo	Recursivo
Operações	152	2075316635
Tempo (s)	< 1	257

Quick Sort

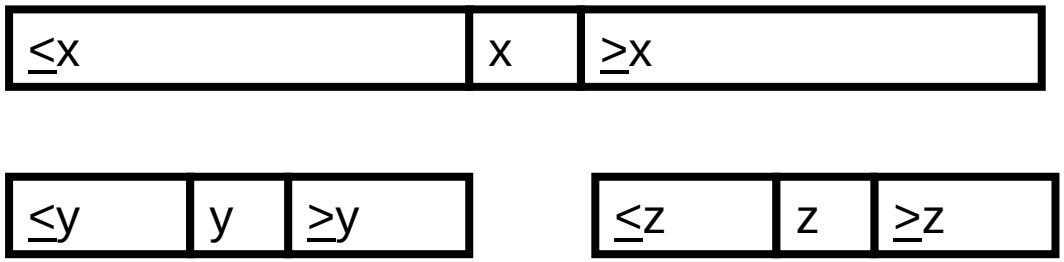
- Também conhecido como ordenação por partição;
- publicado por Hoare, em 1961;
- faz trocas entre elementos mais distantes entre si na lista;
- sequência reversa: apenas n/2 trocas;
- realiza trocas sobre um elemento (pivô) que particiona a lista de maneira conveniente.

Quick Sort

Escolher, de forma aleatória, um elemento **x** da sequência $v_0, v_1, \ldots, v_{n-1}$, de **n** elementos;

Percorrer sequência deixando os valores menores que **x** à sua esquerda e os maiores à sua direita;

Dessa maneira, teremos:



Repetir o processo de **partição** nas duas listas geradas, até que as novas listas possuam apenas um elemento;

A lista é dividida em listas menores e, depois, as listas menores vão aumentando, já ordenadas...

Quick Sort Recursivo

```
void quicksort(int *v, int l, int r) {
   int pivo, novoPivo;
   /* se a lista possui 2 ou mais elementos */
   if (l < r) {
      /* escolher o índice do meio como pivô */
      pivo = (l + r) / 2;
      novoPivo = particionar(v, l, r, pivo);
      /* ordenação recursiva dos subvetores */
      quicksort(v, l, novoPivo - 1);
      quicksort(v, novoPivo + 1, r);
   }
}
```

Quick Sort Recursivo

```
int particionar(int *v, int l, int r, int p) {
   int i, j;
   int valor_pivo = v[p];
   trocar(&v[p], &v[r]);
   j = l;
   for (i = l; i <= r - 1; i++) {
      if (v[i] <= valor_pivo) {
         trocar(&v[i], &v[j]);
         j++;
      }
   }
   trocar(&v[j], &v[r]);
   return j;
}
```

Método da Bissecção Recursivo

- Calcular a raiz da função f(x) = sen(x – 0,1234).
- Sabe-se que a raiz pertence ao intervalo [0; 0,2].
- Implementar o método da bissecção recursivo.
- Considere 10^{-6} como precisão da resposta.