

# MAC0122 Princípios de Desenvolvimento de Algoritmos

Edição 2014

# AULA 1

# Administração

Página da disciplina:

<http://grauna.ime.usp.br/>

- ▶ aulas
- ▶ exercícios-programa
- ▶ fórum: **perguntem, respondam, ...**
- ▶ material: **brinquem com os programas**
- ▶ ...

# Administração

Página da disciplina:

<http://grauna.ime.usp.br/>

- ▶ aulas
- ▶ exercícios-programa
- ▶ fórum: **perguntem, respondam, ...**
- ▶ material: **brinquem com os programas**
- ▶ ...

**Exercício programa 1:** disponível na página

## Livros

Nossa referência básica é o livro

*PF = Paulo Feofiloff,*  
*Algoritmos em linguagem C,*



Este livro é baseado no material do sítio

*Projeto de Algoritmos em C.*

Outros livros são

*S = Robert Sedgwick,*  
*Algorithms in C, vol. 1*

*SW = Robert Sedgwick and Kevin Wayne,*  
*Algorithms*

# Onde você se meteu. . .

*Blue Pill or Red Pill - The Matrix*

Apresentação de **MAC0122** no YouTube:

<https://www.youtube.com/watch?v=OGNTReARNL4>.

**MAC0122** é uma disciplina introdutória em:

- ▶ projeto, correção e eficiência de algoritmos e
- ▶ estruturas de dados

# MAC0122

MAC0122 combina técnicas de

- ▶ programação
- ▶ correção de algoritmos (relações invariantes)
- ▶ análise da eficiência de algoritmos e
- ▶ estruturas de dados elementares

que nasceram de aplicações cotidianas em ciência da computação.

# Pré-requisitos

O pré-requisito oficial de **MAC0122** é

- ▶ **MAC2166** Introdução à Computação.



# Principais tópicos

Alguns dos tópicos de **MAC0122** são:

- ▶ recursão;
- ▶ busca em um vetor;
- ▶ busca (binária) em vetor ordenado;
- ▶ listas encadeadas;
- ▶ listas lineares: filas e pilhas;
- ▶ algoritmos de enumeração;
- ▶ busca de palavras em um texto;
- ▶ algoritmos de ordenação: bubblesort, heapsort, mergesort, . . . ;

Tudo isso regado a muita **análise de eficiência de algoritmos e invariantes**.

# Localização

MAC0122 é um primeiro passo na direção de

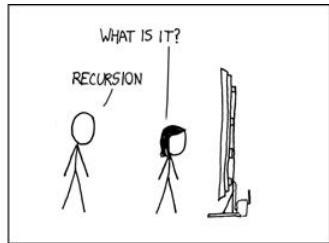
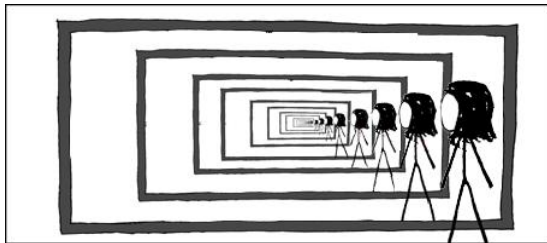
- ▶ Algoritmos
- ▶ Estruturas de Dados

Várias outras disciplina se apoiam em MAC0122.

# Pausa para nossos comerciais

- ▶ XVIII Maratona de Programação: 16 de agosto  
<http://www.ime.usp.br/~cef/XVIIImaratona/>

# Recursão



Fonte: <http://xkcdsw.com/1105>

PF 2.1, 2.2, 2.3 S 5.1

<http://www.ime.usp.br/~pf/algoritmos/aulas/recu.html>

# Recursão

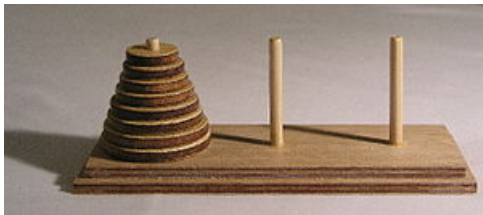
“To understand recursion, we must first understand recursion.”

–folclore

“Para fazer uma função recursivo é preciso ter fé.”

–Siang Wu Song

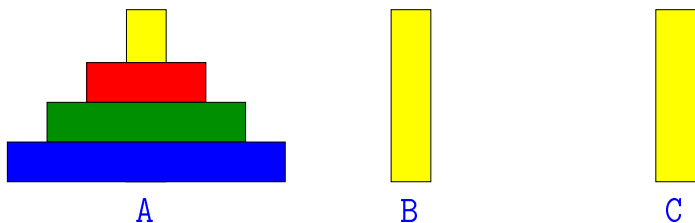
# Torres de Hanoi



Fonte: <http://commons.wikimedia.org/>  
Licensed under Creative Commons Attribution  
Share Alike 3.0 via Wikimedia Commons

[http://en.wikipedia.org/wiki/Hanoi\\_tower](http://en.wikipedia.org/wiki/Hanoi_tower)

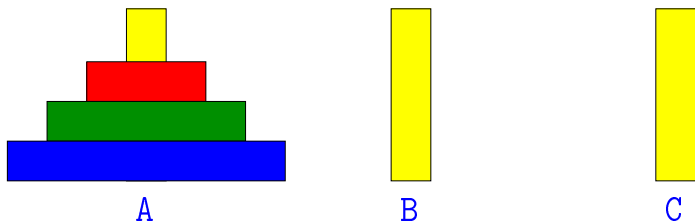
# Torres de Hanoi



Desejamos transferir  $n$  discos do pino A para o pino C usando o pino B como auxiliar e repetando as regras:

- ▶ podemos mover apenas um disco por vez;
- ▶ nunca um disco de diâmetro maior poderá ser colocado sobre um disco de diâmetro menor.

# Torres de Hanoi

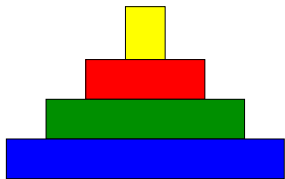


Denotaremos por  $\text{Hanoi}(n, A, B, C)$  o problema de transferir  $n$  discos do pino  $A$  para o pino  $C$  usando o pino  $B$  como auxiliar

Como resolver  $\text{Hanoi}(n, A, B, C)$ ?



## Idéia



A



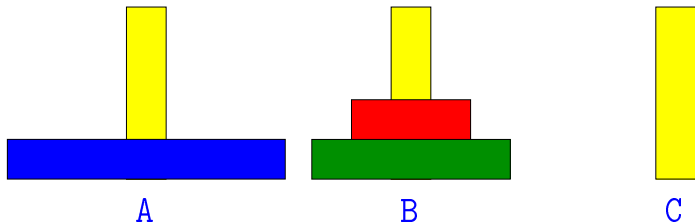
B



C

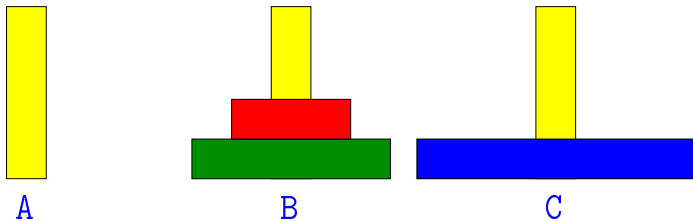
Posso não saber qual o primeiro movimento, mas é fácil saber qual é o **movimento do meio**.

## Idéia



Posso não saber qual o primeiro movimento, mas é fácil saber qual é o **movimento do meio**.

## Idéia



Posso não saber qual o primeiro movimento, mas é fácil saber qual é o **movimento do meio**.

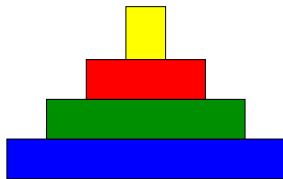
## Idéia



A



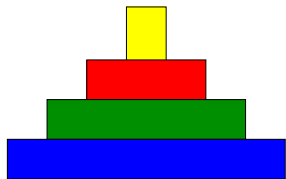
B



C

Posso não saber qual o primeiro movimento, mas é fácil saber qual é o **movimento do meio**.

# Solução



A



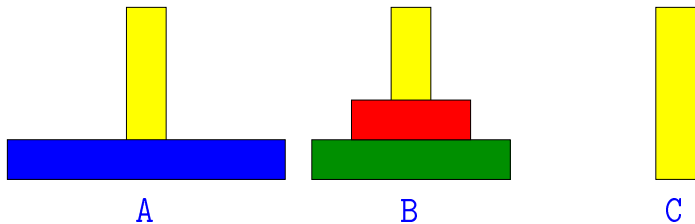
B



C

Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

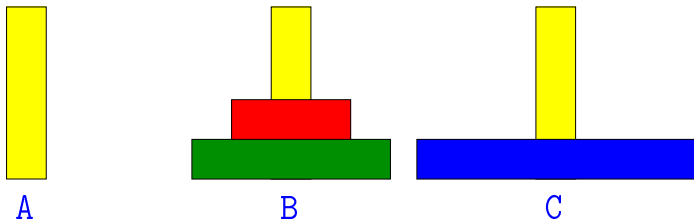
# Solução



Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(\underline{n-1}, A, C, B)$

# Solução



Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(n-1, A, C, B)$
2. mover o disco  $n$  de  $A$  para  $C$

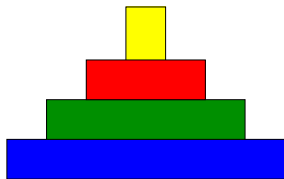
# Solução



A



B



C

Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(\underline{n-1}, A, C, B)$
2. mover o disco  $n$  de  $A$  para  $C$
3. resolver  $\text{Hanoi}(\underline{n-1}, B, A, C)$



## Solução

Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(\underline{n-1}, A, C, B)$
2. mover o disco  $n$  de  $A$  para  $C$
3. resolver  $\text{Hanoi}(\underline{n-1}, B, A, C)$

E daí?

## Solução

Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(n-1, A, C, B)$
2. mover o disco  $n$  de  $A$  para  $C$
3. resolver  $\text{Hanoi}(n-1, B, A, C)$

E daí?

Reduzimos o problema com  $n$  discos para 2 problemas com  $n-1$  disco!

## Solução

Para resolver  $\text{Hanoi}(n, A, B, C)$  basta:

1. resolver  $\text{Hanoi}(n-1, A, C, B)$
2. mover o disco  $n$  de  $A$  para  $C$
3. resolver  $\text{Hanoi}(n-1, B, A, C)$

E daí?

Reduzimos o problema com  $n$  discos para 2 problemas com  $n-1$  disco!

Paramos de reduzir quando soubermos resolver o problema. Por exemplo, sabemos resolver

$\text{Hanoi}(0, \dots, \dots, \dots)$

## Função que resolve o problema

```
void
hanoi(int n, char origem, char auxiliar,
      char destino)
{
    if (n > 0)
    {
        hanoi(n-1, origem, destino, auxiliar);
        printf("mova disco %d de %c para %c.\n",
              n, origem, destino);
        hanoi(n-1, auxiliar, origem, destino);
    }
}
```

Primeira chamada: `hanoi(n, 'A', 'B', 'C');`

hanoi(3, 'A', 'B', 'C')

- 1: mova o disco 1 do pino A para o pino C.
- 2: mova o disco 2 do pino A para o pino B.
- 3: mova o disco 1 do pino C para o pino B.
- 4: mova o disco 3 do pino A para o pino C.
- 5: mova o disco 1 do pino B para o pino A.
- 6: mova o disco 2 do pino B para o pino C.
- 7: mova o disco 1 do pino A para o pino C.

# hanoi(4, 'A', 'B', 'C')

- 1: mova o disco 1 do pino A para o pino B.
- 2: mova o disco 2 do pino A para o pino C.
- 3: mova o disco 1 do pino B para o pino C.
- 4: mova o disco 3 do pino A para o pino B.
- 5: mova o disco 1 do pino C para o pino A.
- 6: mova o disco 2 do pino C para o pino B.
- 7: mova o disco 1 do pino A para o pino B.
- 8: mova o disco 4 do pino A para o pino C.
- 9: mova o disco 1 do pino B para o pino C.
- 10: mova o disco 2 do pino B para o pino A.
- 11: mova o disco 1 do pino C para o pino A.
- 12: mova o disco 3 do pino B para o pino C.
- 13: mova o disco 1 do pino A para o pino B.
- 14: mova o disco 2 do pino A para o pino C.
- 15: mova o disco 1 do pino B para o pino C.

# hanoi(7, 'A', 'B', 'C')

1: mova o disco 1 do pino A para 44pimovA. o disco 3 do pino B para85:pimovA o disco 1 do pino C para o pin  
2: mova o disco 2 do pino A para 45pimovA. o disco 1 do pino C para86:pimovA o disco 4 do pino C para o pin  
3: mova o disco 1 do pino C para 46pimovA. o disco 2 do pino C para88:pimovA o disco 1 do pino B para o pin  
4: mova o disco 3 do pino A para 47pimovA. o disco 1 do pino B para90:pimovA o disco 2 do pino B para o pin  
5: mova o disco 1 do pino B para 48pimovA. o disco 5 do pino C para94:pimovA o disco 1 do pino A para o pin  
6: mova o disco 2 do pino B para 49pimovA. o disco 1 do pino A para92:pimovA o disco 3 do pino B para o pin  
7: mova o disco 1 do pino A para 50pimovA. o disco 2 do pino A para98:pimovA o disco 1 do pino C para o pin  
8: mova o disco 4 do pino A para 51pimovA. o disco 1 do pino C para94:pimovA o disco 2 do pino C para o pin  
9: mova o disco 1 do pino C para 52pimovA. o disco 3 do pino A para95:pimovA o disco 1 do pino B para o pin  
10: mova o disco 2 do pino C para58:pimovA o disco 1 do pino B para96:pimovA o disco 6 do pino B para o pin  
11: mova o disco 1 do pino B para54:pimovA o disco 2 do pino B para97:pimovA o disco 1 do pino A para o pin  
12: mova o disco 3 do pino C para55:pimovA o disco 1 do pino A para98:pimovA o disco 2 do pino A para o pin  
13: mova o disco 1 do pino A para56:pimovA o disco 4 do pino A para99:pimovA o disco 1 do pino C para o pin  
14: mova o disco 2 do pino A para57:pimovA o disco 1 do pino C para100pimovA. o disco 3 do pino A para o pi  
15: mova o disco 1 do pino C para58:pimovA o disco 2 do pino C para101pimovA. o disco 1 do pino B para o pi  
16: mova o disco 5 do pino A para59:pimovA o disco 1 do pino B para102pimovA. o disco 2 do pino B para o pi  
17: mova o disco 1 do pino B para60:pimovA o disco 3 do pino C para103pimovA. o disco 1 do pino A para o pi  
18: mova o disco 2 do pino B para64:pimovA o disco 1 do pino A para104pimovA. o disco 4 do pino A para o pi  
19: mova o disco 1 do pino A para62:pimovA o disco 2 do pino A para105pimovA. o disco 1 do pino C para o pi  
20: mova o disco 3 do pino B para68:pimovA o disco 1 do pino C para106pimovA. o disco 2 do pino C para o pi  
21: mova o disco 1 do pino C para64:pimovA o disco 7 do pino A para107pimovA. o disco 1 do pino B para o pi  
22: mova o disco 2 do pino C para65:pimovA o disco 1 do pino B para108pimovA. o disco 3 do pino C para o pi  
23: mova o disco 1 do pino B para66:pimovA o disco 2 do pino B para109pimovA. o disco 1 do pino A para o pi  
24: mova o disco 4 do pino B para67:pimovA o disco 1 do pino A para1040pimovA. o disco 2 do pino A para o pi  
25: mova o disco 1 do pino A para68:pimovA o disco 3 do pino B para1041pimovA. o disco 1 do pino C para o pi  
26: mova o disco 2 do pino A para69:pimovA o disco 1 do pino C para1042pimovA. o disco 5 do pino A para o pi  
27: mova o disco 1 do pino C para70:pimovA o disco 2 do pino C para1043pimovA. o disco 1 do pino B para o pi  
28: mova o disco 3 do pino A para74:pimovA o disco 1 do pino B para1044pimovA. o disco 2 do pino B para o pi  
29: mova o disco 1 do pino B para72:pimovA o disco 4 do pino B para1045pimovA. o disco 1 do pino A para o pi  
30: mova o disco 2 do pino B para78:pimovA o disco 1 do pino A para1046pimovA. o disco 3 do pino B para o pi  
31: mova o disco 1 do pino A para74:pimovA o disco 2 do pino A para1047pimovA. o disco 1 do pino C para o pi  
32: mova o disco 6 do pino A para75:pimovA o disco 1 do pino C para1048pimovA. o disco 2 do pino C para o pi  
33: mova o disco 1 do pino C para76:pimovA o disco 3 do pino A para1049pimovA. o disco 1 do pino B para o pi  
34: mova o disco 2 do pino C para78:pimovA o disco 1 do pino B para1050pimovA. o disco 4 do pino B para o pi  
35: mova o disco 1 do pino B para78:pimovA o disco 2 do pino B para1081pimovA. o disco 1 do pino A para o pi

# Recursão

A resolução recursiva de um problema tem tipicamente a seguinte estrutura:

**se** a instância em questão é “pequena”  
  **resolva-a diretamente**  
  (use força bruta se necessário);

**senão**  
  reduza-a a uma instância “menor”  
  do **mesmo problema**,  
  **aplique o método à instância menor** e  
  volte à instância original.



# Curiosidades

Veja “Debugging recursive code” em :

<http://devopsreactions.tumblr.com/>

## Fatorial recursivo

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

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

fatorial(10)

fatorial(10)

fatorial(9)

fatorial(8)

fatorial(7)

fatorial(6)

fatorial(5)

fatorial(4)

fatorial(3)

fatorial(2)

fatorial(1)

fatorial(0)

fatorial de 10 e' 3628800.

# Diagramas de execução

fatorial(3)

n

3

fatorial(2)

n

2

fatorial(1)

n

1

fatorial(1)

n

0

return 1

return n \* fatorial(0) = 1 \* 1

return n \* fatorial(1) = 2 \* 1 = 2

return n \* fatorial(2) = 3 \* 2 = 6

```
hanoi(2, 'A', 'B', 'C')
```

```
hanoi(1, 'A', 'C', 'B')
```

```
hanoi(0, 'A', 'B', 'C')
```

1: mova o disco 1 do pino A para o pino B.

```
hanoi(0, 'B', 'A', 'B')
```

2: mova o disco 2 do pino A para o pino C.

```
hanoi(1, 'B', 'A', 'C')
```

```
hanoi(0, 'B', 'C', 'A')
```

3: mova o disco 1 do pino B para o pino C.

```
hanoi(0, 'A', 'B', 'C')
```

## Fatorial iterativo

```
long
fatorial(long n)
{
    int i, ifat;

    ifat = 1;
    for(i = 1; /*1*/ i <= n; i++)
        ifat *= i;

    return ifat;
}
```

Em /\*1\*/ vale que `ifat == (i-1)!`