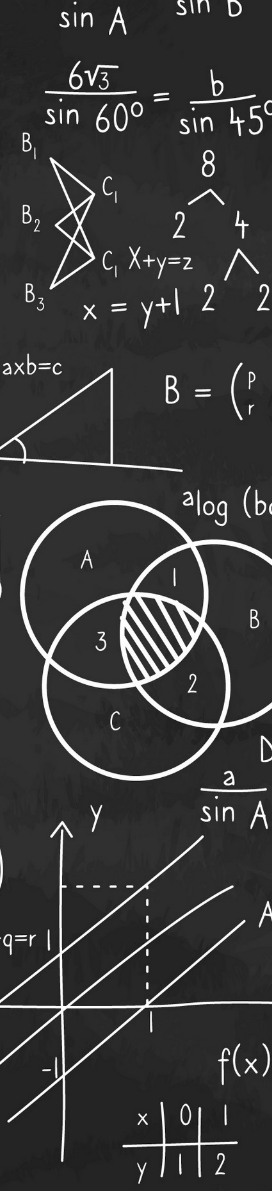


Matemática Discreta 2

Aula 02

Cristiane Loesch

cristiane.costa@unb.br

Brasília
2025

INDUÇÃO MATEMÁTICA (forte)

EXERCÍCIO:

1) Prove que

$$u_n = 2^n + (-1)^n$$

dado

$$u_n = 1, 5, \ldots, u_{n-1} + 2u_{(n-2)} \quad , \quad n > 2$$

2) Prove que

$$2 + 4 + 6 + 8 + \ldots + 2n = n(n+1) \quad , \quad \forall\, n \in \mathbb{N}$$

3) Prove que

$$2^n > n^2 \quad , \quad \forall\, n \geq 5 \quad , \quad n \in \mathbb{N}$$

RECORRÊNCIA LINEAR

EXERCÍCIO:

4) Escreva a fórmula fechada da recorrência

a) $\begin{cases} a_n = 2\,a_{n-1} \\ a_0 = 5 \end{cases}$

b) $\begin{cases} a_n = a_{n-1} + 6 \\ a_0 = 4 \end{cases}$

SOLUÇÃO DE PROBLEMAS

INDUÇÃO (FRACA OU FORTE)

→ Passo Básico
→ Hipótese Indutiva
→ Passo Indutivo

MATEMÁTICA

RECURSÃO

→ Passo Base → { caso base / caso geral }
→ Passo Recursivo

→ APLICAÇÕES
- equações aritméticas (ex. recorrência, algoritmo de Euclides, etc)
- algoritmos recursivos (ex. Torre de Hanoi)
- etc

SOLUÇÃO DE PROBLEMAS

MATEMÁTICA

INDUÇÃO (FRACA OU FORTE)

→ Passo Básico
→ Hipótese Indutiva
→ Passo Indutivo

RECURSÃO

→ Passo Base → caso base / caso geral
→ Passo Recursivo

problemas computacionais em que cada instância do problema contém uma instância menor do mesmo problema, dizem-se problemas de estrutura recursiva

→ APLICAÇÕES
- equações aritméticas (ex. recorrência, algoritmo de Euclides, etc)
- algoritmos recursivos (ex. Torre de Hanoi)
- etc

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

→ Método de solução de problemas que envolve “quebrar” um problema em subproblemas menores e menores até chegar a um problema pequeno o suficiente para que ele possa ser resolvido trivialmente.

→ definição de um objeto em função de si mesmo

→ função recursiva:

- função que chama a si própria (recursão direta) ou, ainda, indiretamente (recursão indireta) para resolver um problema.

Obs: algoritmos convencionais podem ser escritos de forma recursiva e vice-versa

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Fonte: Z. Dias (2023)

Solução do problema

1) Caso Base

2) Caso Geral

3) Função Recursiva

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

PROBLEMA

MUITO SIMPLES SIMPLES COMPLEXO

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

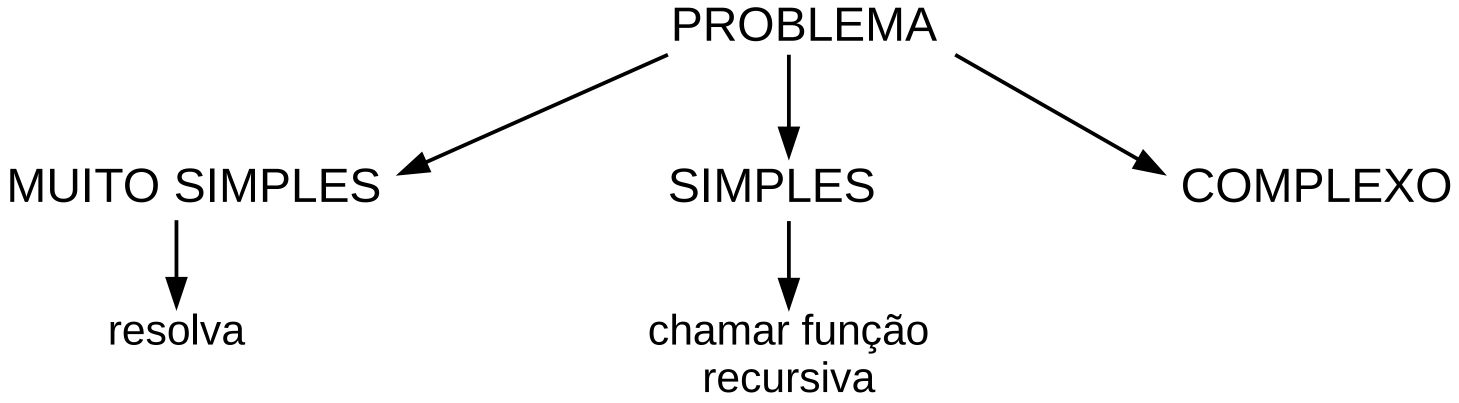
PROBLEMA

MUITO SIMPLES SIMPLES COMPLEXO

resolva

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

PROBLEMA

MUITO SIMPLES

SIMPLES

COMPLEXO

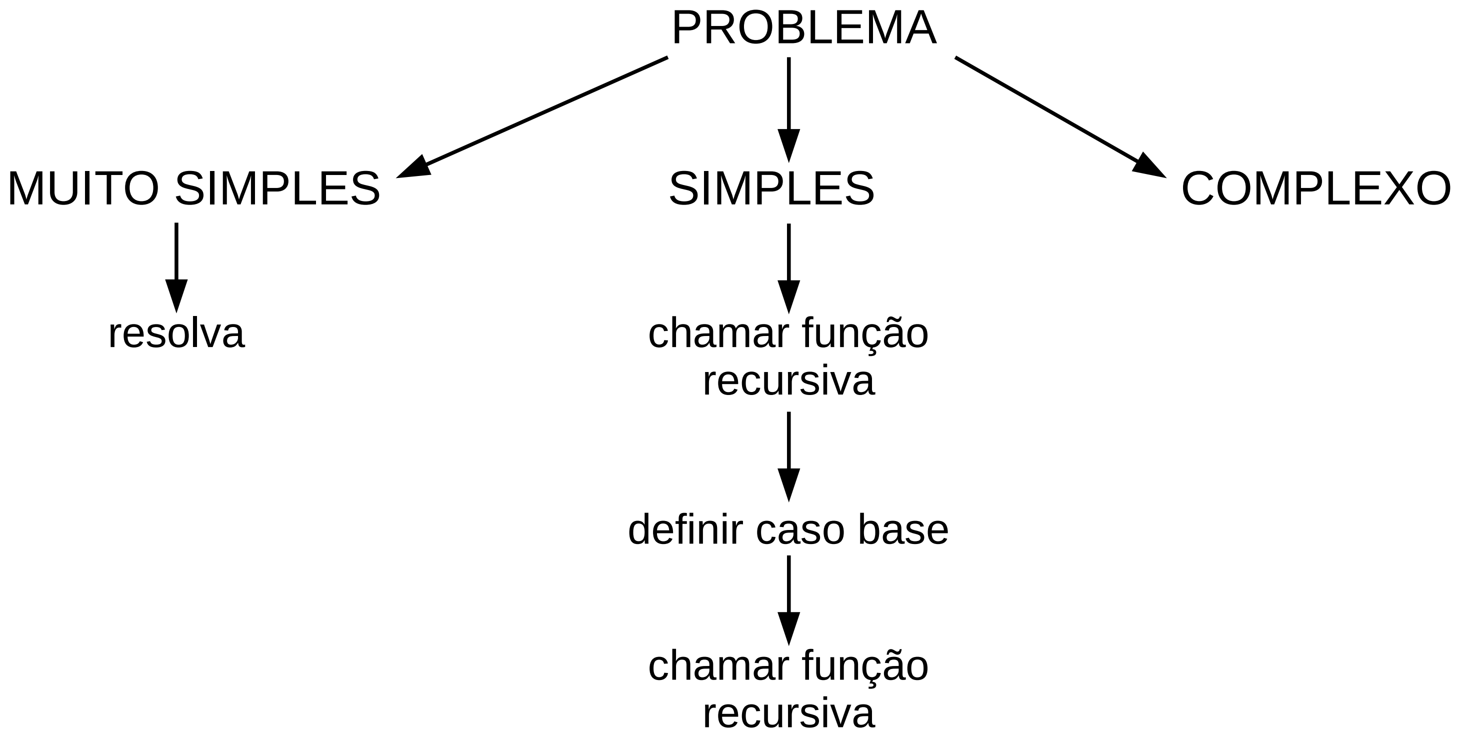
resolva

chamar função recursiva

definir caso base

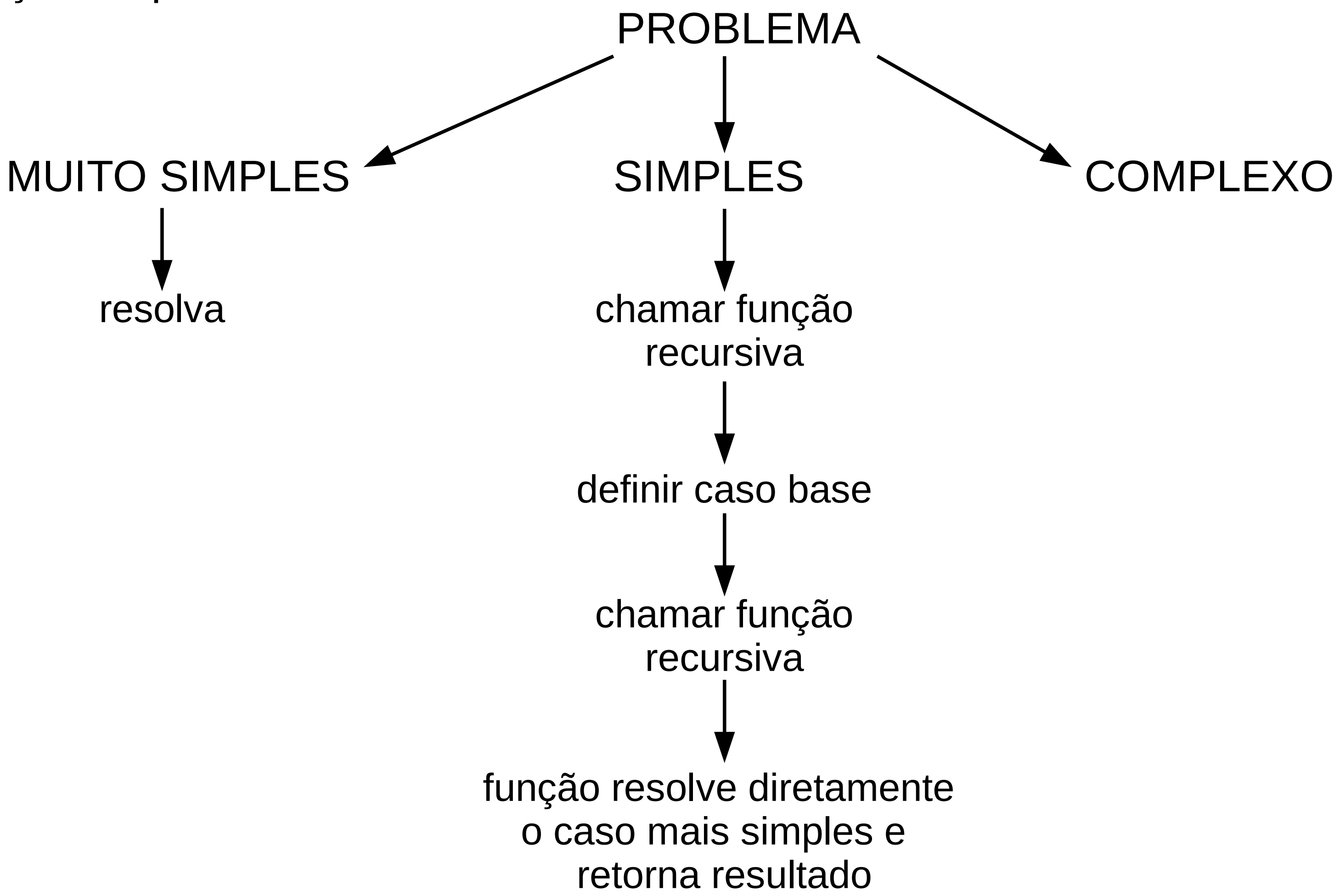
RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

PROBLEMA

- MUITO SIMPLES → resolva
- SIMPLES → chamar função recursiva → definir caso base → chamar função recursiva → função resolve diretamente o caso mais simples e retorna resultado
- COMPLEXO → Problema a resolver

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

PROBLEMA

MUITO SIMPLES

resolva

SIMPLES

chamar função recursiva

definir caso base

chamar função recursiva

função resolve diretamente o caso mais simples e retorna resultado

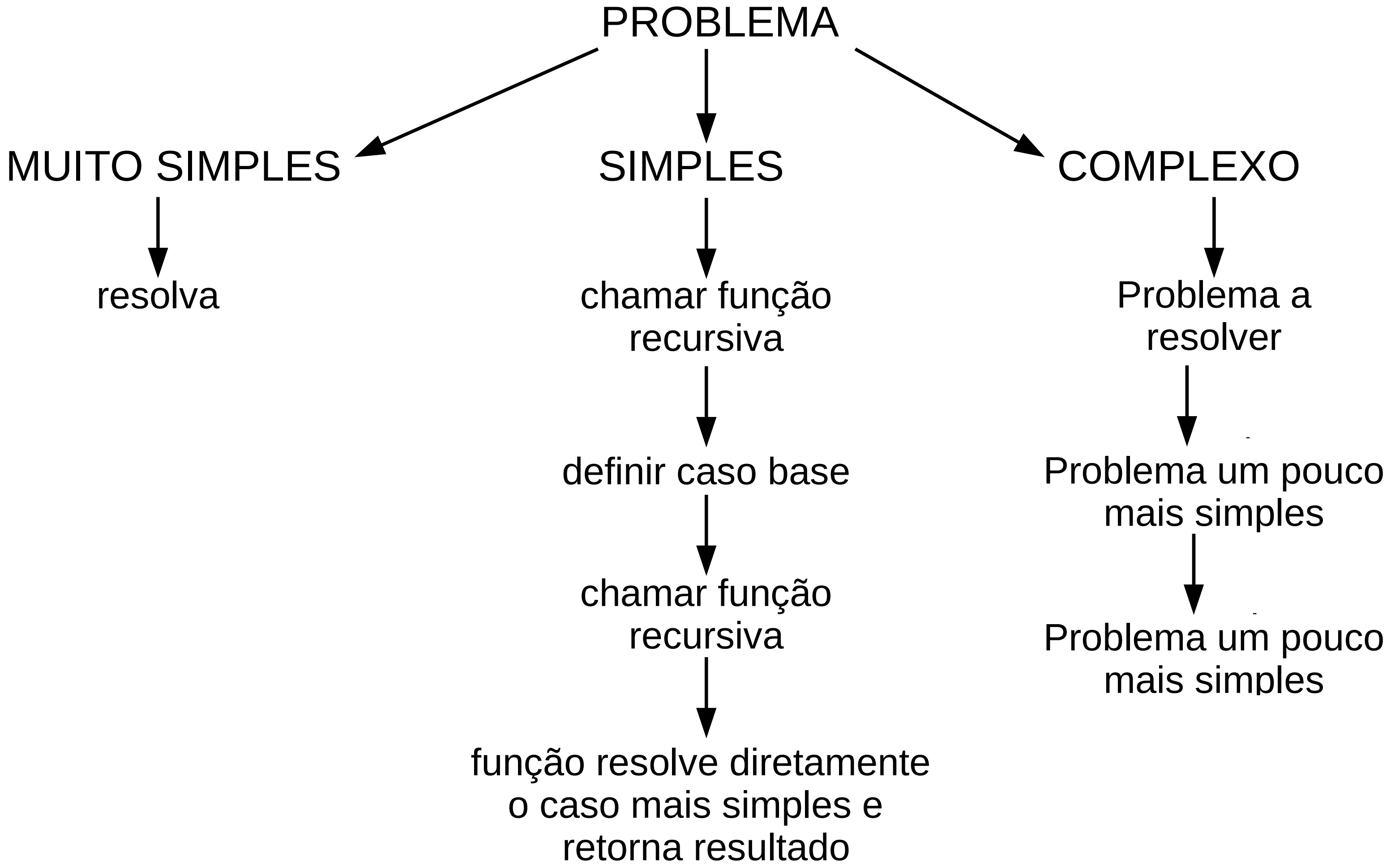
COMPLEXO

Problema a resolver

Problema um pouco mais simples

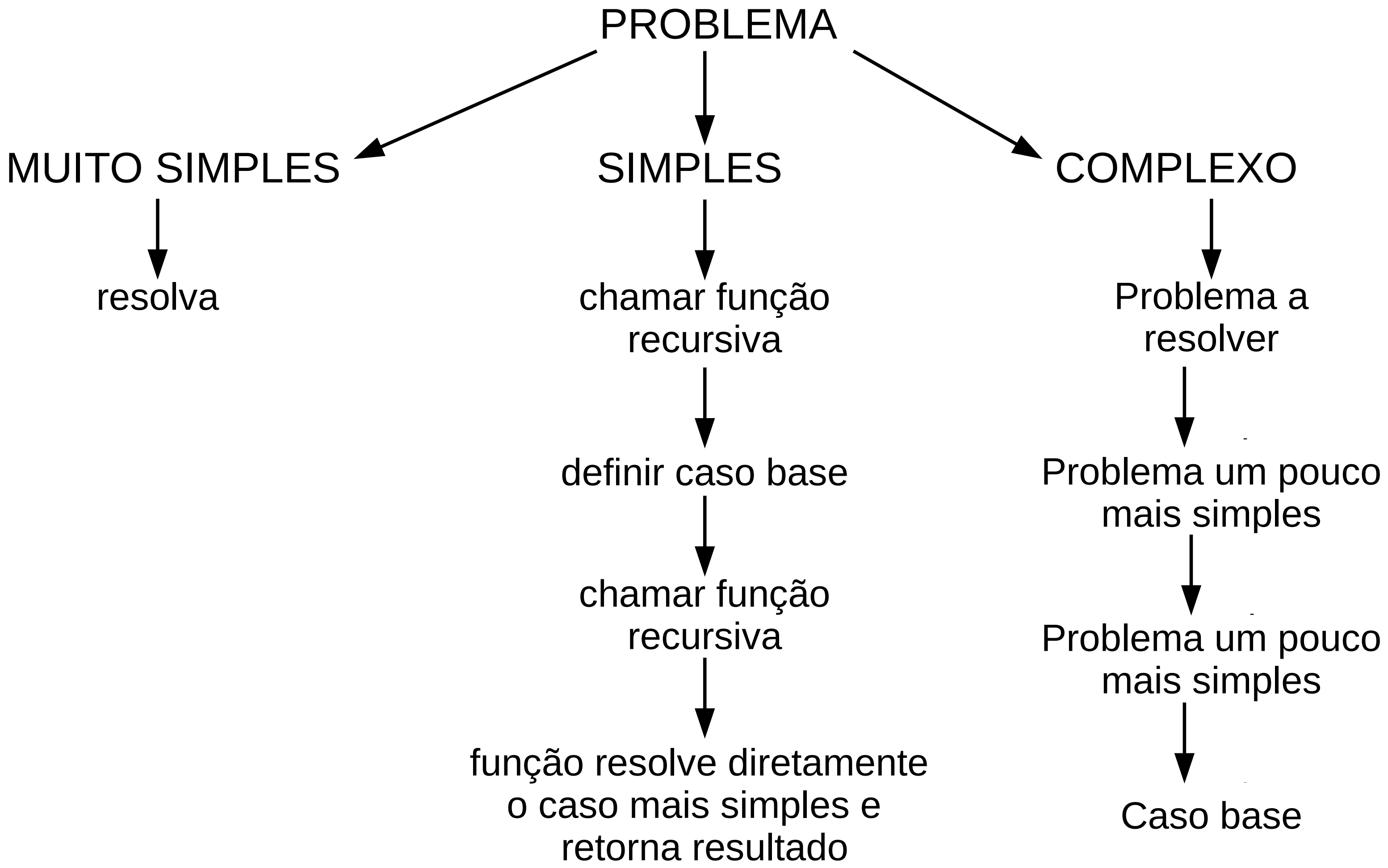
RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema

PROBLEMA

MUITO SIMPLES

resolva

SIMPLES

chamar função recursiva

definir caso base

chamar função recursiva

função resolve diretamente o caso mais simples e retorna resultado

COMPLEXO

Problema a resolver

Problema um pouco mais simples

Problema um pouco mais simples

Caso base

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva $\rightarrow$

Função divide o problema em duas partes conceituais

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

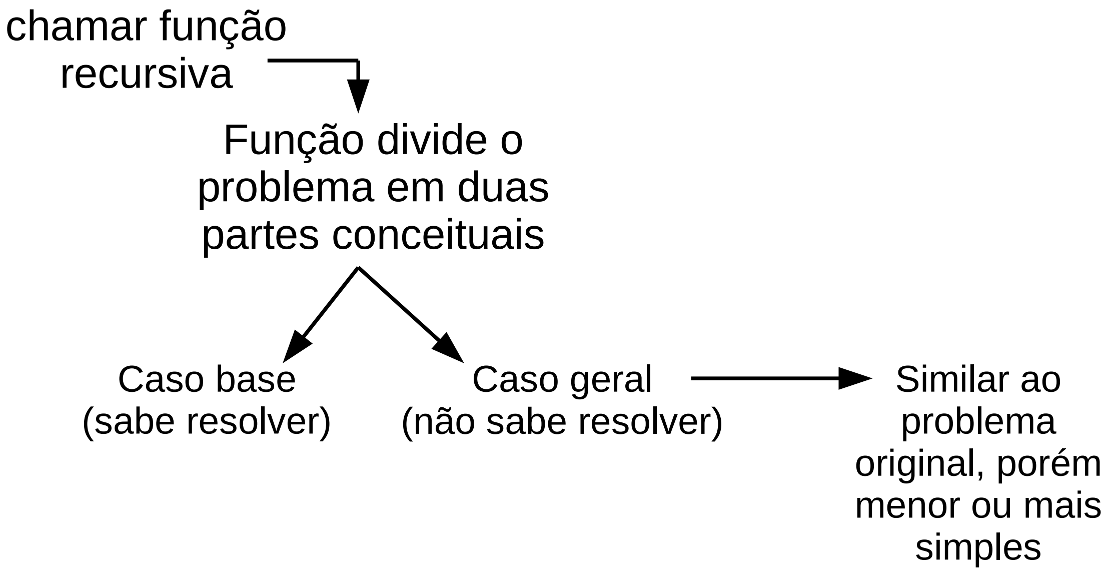
Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS



RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

Similar ao problema original, porém menor ou mais simples

função se invoca para tentar solucionar (Passo recursivo)

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

Similar ao problema original, porém menor ou mais simples

função se invoca para tentar solucionar (Passo recursivo)

Passo recursivo é executado, não sabe resolver, simplifica novamente

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

Similar ao problema original, porém menor ou mais simples

função se invoca para tentar solucionar (Passo

Passo recursivo é executado, não sabe resolver, simplifica

função se invoca novamente para tentar solucionar

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

Similar ao problema original, porém menor ou mais simples

função se invoca para tentar solucionar (Passo recursivo)

Passo recursivo é executado, não sabe resolver, simplifica

função se invoca novamente para tentar solucionar

Processo de simplificação se repete até chegar ao caso base

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

Solução do problema: PROBLEMAS MAIS COMPLEXOS

chamar função recursiva

Função divide o problema em duas partes conceituais

Caso base (sabe resolver)

Caso geral (não sabe resolver)

Similar ao problema original, porém menor ou mais simples

função se invoca para tentar solucionar (Passo recursivo)

Passo recursivo é executado, não sabe resolver, simplifica novamente

função se invoca novamente para tentar solucionar

Processo de simplificação se repete até chegar ao caso base

Caso base é resolvido, Função retorna o valor e utiliza-o para solucionar os casos um pouco mais complexos

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\,P(n-1)\ ,\quad n>1$

$P(5)$?

$P(5) = 2\,P(4)$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\ P(n-1)\ ,\ \ n>1$

$P(5)?$

$P(5) = 2\ P(4)$

$P(4) = 2\ P(3)$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\,P(4)$

$P(4) = 2\,P(3)$

$P(3) = 2\,P(2)$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\ P(n-1)\ ,\ \ n>1$

$P(5)?$

$P(5) = 2\ P(4)$
$P(4) = 2\ P(3)$
$P(3) = 2\ P(2)$
$P(2) = 2\ P(1)$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\,P(4)$
$P(4) = 2\,P(3)$
$P(3) = 2\,P(2)$
$P(2) = 2\,P(1)$
$P(1) = 2$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\ P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\ P(4)$

$P(4) = 2\ P(3)$

$P(3) = 2\ P(2)$

$P(2) = 2\ P(1)$

$P(1) = 2$

Caso Base $\boxed{P(1) = 2}$

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
P(1) = 2
P(n) = 2 P(n-1) , n>1

P(5)?

P(5) = 2 P(4)
P(4) = 2 P(3)
P(3) = 2 P(2)
P(2) = 2 P(1)
P(1) = 2

P(2) = 2 P(1)

$\boxed{2}$

Caso Base

P(1) = 2

Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:

$P(1) = 2$

$P(n) = 2\ P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\ P(4)$

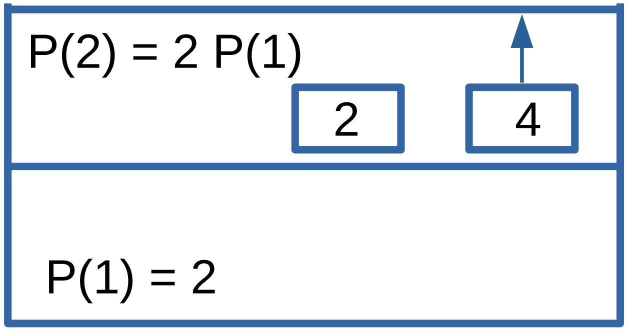
$P(4) = 2\ P(3)$

$P(3) = 2\ P(2)$

$P(2) = 2\ P(1)$

$P(1) = 2$

Caso Base



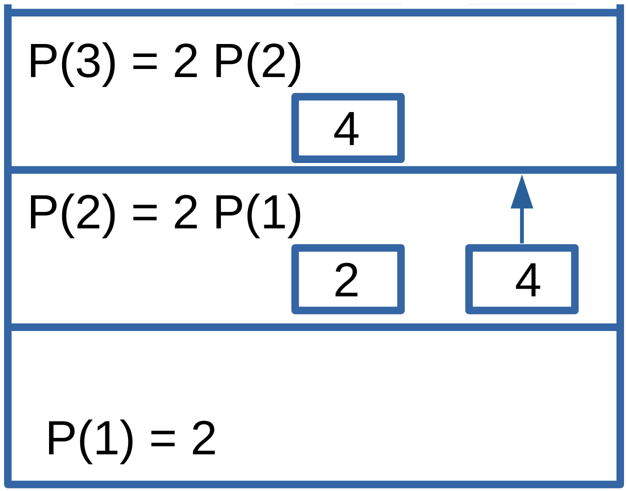
Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\,P(4)$
$P(4) = 2\,P(3)$
$P(3) = 2\,P(2)$
$P(2) = 2\,P(1)$
$P(1) = 2$



Caso Base

Obs: computacionalmente funciona como uma pilha

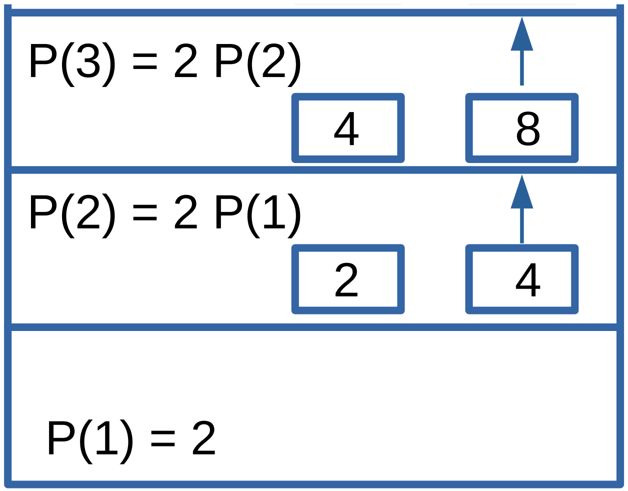
RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
P(1) = 2
P(n) = 2 P(n-1) , n>1

P(5)?

P(5) = 2 P(4)
P(4) = 2 P(3)
P(3) = 2 P(2)
P(2) = 2 P(1)
P(1) = 2

Caso Base



Obs: computacionalmente funciona como uma pilha

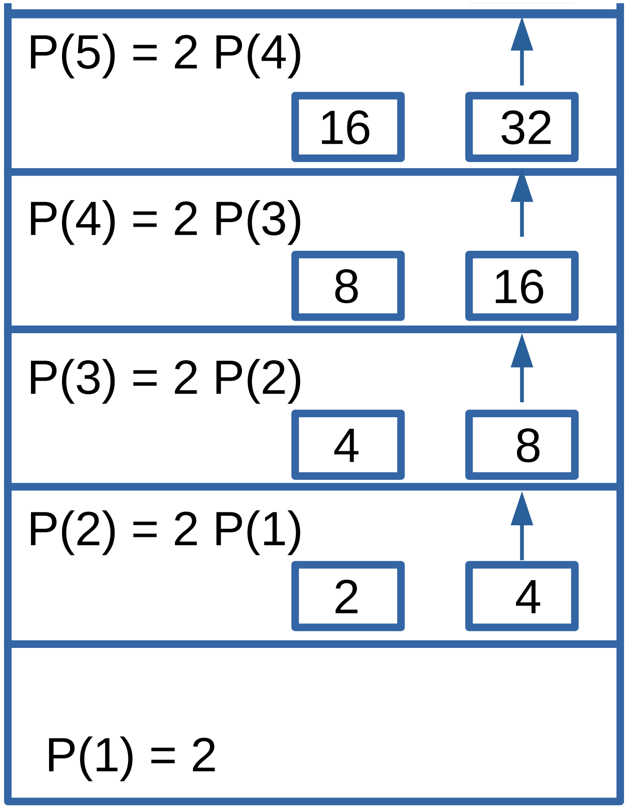
RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\,P(4)$
$P(4) = 2\,P(3)$
$P(3) = 2\,P(2)$
$P(2) = 2\,P(1)$
$P(1) = 2$

Caso Base



Obs: computacionalmente funciona como uma pilha

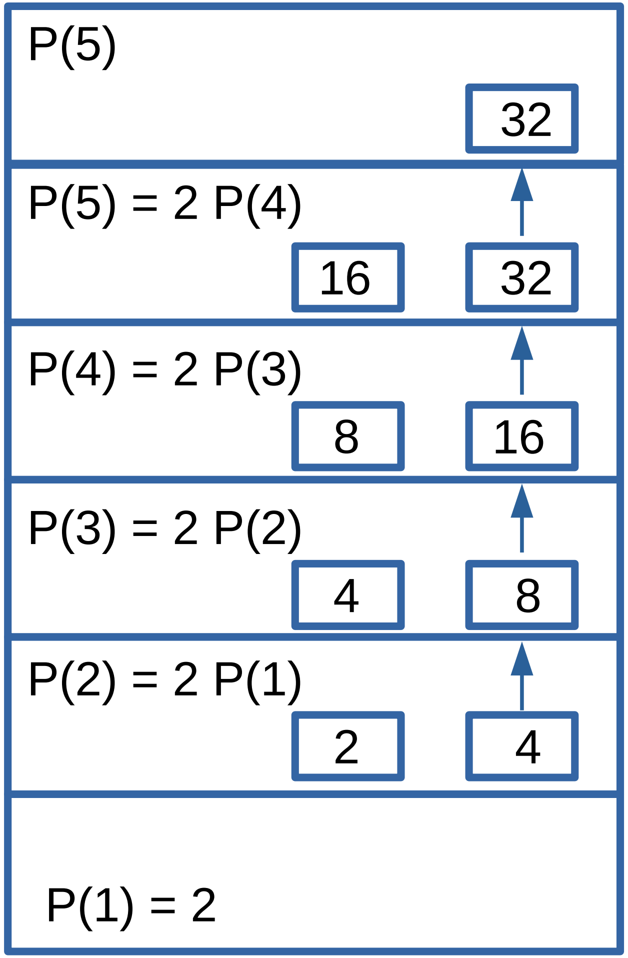
RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\ P(n-1)\ ,\ \ n>1$

$P(5)$?

$P(5) = 2\ P(4)$
$P(4) = 2\ P(3)$
$P(3) = 2\ P(2)$
$P(2) = 2\ P(1)$
$P(1) = 2$

Caso Base



Obs: computacionalmente funciona como uma pilha

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO:
$P(1) = 2$
$P(n) = 2\,P(n-1)\ ,\ \ n>1$

$P(5)$?

$$P(5) = 2\,P(4) = 32 = 2^5$$
$$P(4) = 2\,P(3) = 16 = 2^4$$
$$P(3) = 2\,P(2) = 8 = 2^3$$
$$P(2) = 2\,P(1) = 4 = 2^2$$
$$P(1) = 2$$

Observe que neste caso, é possível conjecturar uma **solução fechada** para a recursão:

$$P(n) = 2^n$$

Tal conjectura pode ser demonstrada por indução matemática.

$$P(1)=2$$
$$2\,P(n-1) = 2^n$$

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

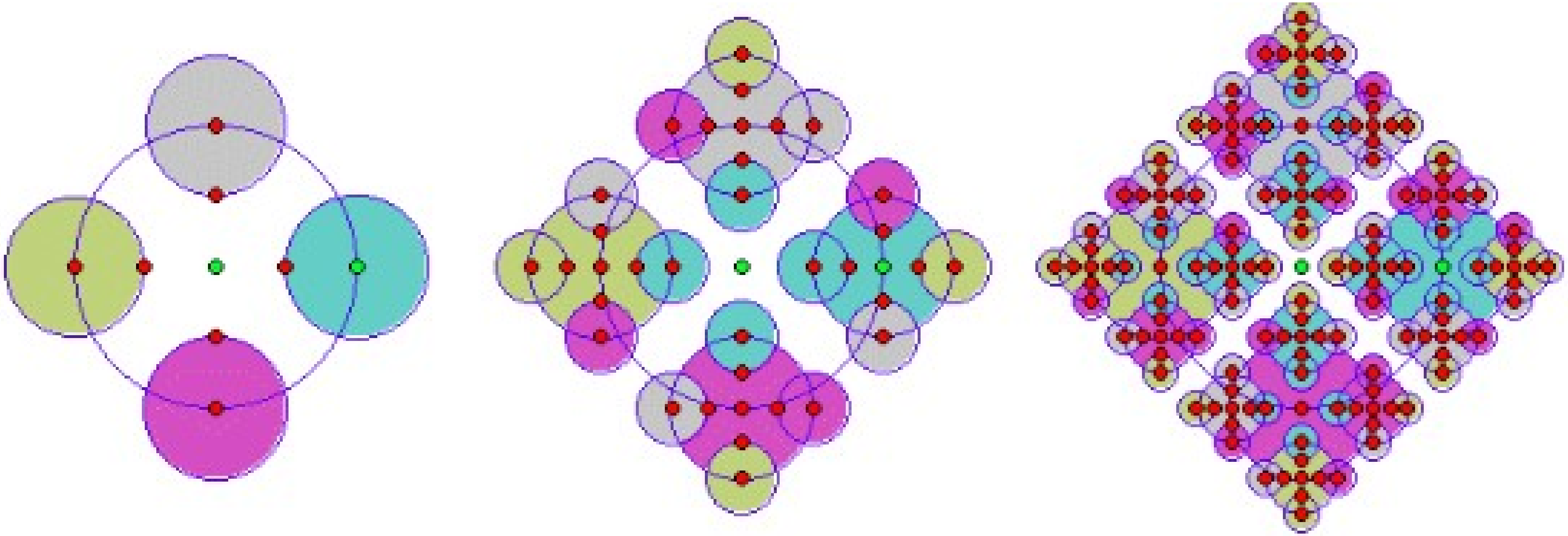


Fig. 1. Representações dos 3 primeiros níveis de recorrência para construir o fractal ***tetra-círculo****.*

https://www.ime.usp.br/~leo/mac2166/2017-1/introducao_recursividade.html#:~:text=Um%20primeiro%20exemplo%20de%20fun%C3%A7%C3%A3o,%2D2)%20x%20…

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO CLÁSSICO:
Definir a função fatorial de forma recursiva

*demonstração em aula

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO CLÁSSICO:
Definir a função fatorial de forma recursiva

Table 4. Factorial n! in Pseudocode

Recursive	Iterative

`procedure factorial(n)`
`begin`
`    if n equals 0 then`
`        return 1`
`    else`
`        return  n * factorial(n-1)`
`    end if`
`end`
	`procedure factorial(n)`
	`begin`
	`    z = 1`
	`    for i = 1 to n do`
	`        z = z * i`
	`    return z`
	`end`

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXERCÍCIO:

1) Escreva a recursão de $P(n) = a^n$, n natural, a real

RECURSÃO OU RECURSIVIDADE MATEMÁTICA

EXEMPLO CLÁSSICO:

Recursive Implementation of the Power Function b^n

Table 5. Power Function b^n in Pseudocode

Recursive	Iterative

`procedure exponentiation(b, n)`
`begin`
`    if n equals 0 then`
`        return 1`
`    else`
`        return  b * exponentiation(n-1)`
`    end if`
`end`
	`procedure exponentiation(b, n)`
	`begin`
	`    z = 1`
	`    for i = 1 to n do`
	`        z = z * b`
	`    return z`
	`end`

DEFINIÇÃO RECURSIVA DE CONJUNTO

Passo base: Especifica-se uma coleção inicial de objetos pertencente ao conjunto.

Passo recursivo: Especificam-se regras para formar novos elementos a partir dos elementos já pertencentes ao conjunto.

A definição recursiva de conjuntos depende da seguinte regra, frequentemente, implícita:

Regra de exclusão: elementos que não podem ser gerados a partir da aplicação do passo base e instâncias do passo indutivo não pertencem ao conjunto

FONTE: Barbosa, H. - UFMG (2020).

DEFINIÇÃO RECURSIVA DE CONJUNTO

EXEMPLO: Seja o conjunto S definido como:

$$\begin{cases} 3 \in S, \\ \text{se } x \in S \text{ e } y \in S, \text{ então } x + y \in S. \end{cases}$$

Determine sua recursividade

FONTE: Barbosa, H. - UFMG (2020).

DEFINIÇÃO RECURSIVA DE ….

1) STRING

EXEMPLO: O conjunto Σ^* de strings sobre um alfabeto

Passo base: $\lambda \in \Sigma^*$ (λ = string vazia).

Passo recursivo: Se $w \in \Sigma^*$ e $x \in \Sigma^* \Rightarrow wx \in \Sigma^*$

(wx representa a string = simbolo x concatenado ao final do prefixo w)

2) SENTENÇAS LÓGICAS

EXEMPLO:

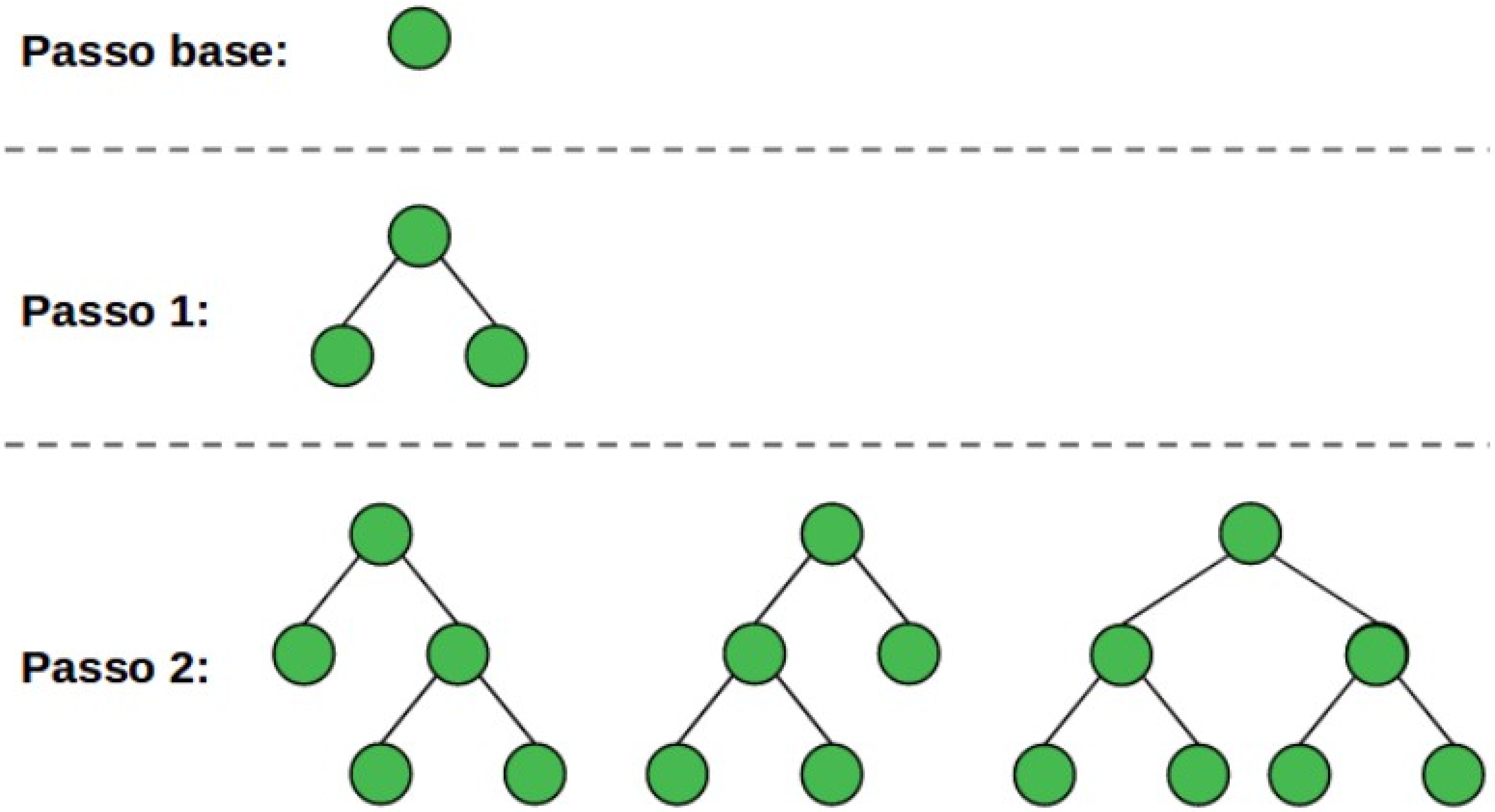
Passo base: T, F e s são sentenças bem formadas, onde s representa uma proposição lógica.

Passo recursivo: Se G e H são sentenças bem formadas, então $(\neg G)$, $(G \wedge H)$, $(G \vee H)$, $(G \rightarrow H)$ e $(G \leftrightarrow H)$ são sentenças bem formadas

FONTE: Barbosa, H. - UFMG (2020).

DEFINIÇÃO RECURSIVA DE ….

3) GRAFOS E ÁRVORES



FONTE: Barbosa, H. - UFMG (2020).

INDUÇÃO ESTRUTURAL

→ conjunto tem definição recursiva

→ propriedades dos elementos do conjunto podem ser demonstradas por recursão

Indução Estrutural :

Passo Base: elementos iniciais do conjunto satisfazem certa propriedade

Hipótese Indutiva: proposição vale para cada um dos elementos usados para construir novos elementos do conjunto

Passo Indutivo: regras de construção de novos elementos preservam tal propriedade

Assim, todos os elementos do conjunto satisfazem a propriedade

INDUÇÃO ESTRUTURAL

EXEMPLO: Seja o conjunto A definido como:

$$\begin{cases} 3 \in A \\ \text{se } x \in A \text{ e } y \in A, \text{ então } x + y \in A \end{cases}$$

Mostre, por indução matemática, que os elementos de A são divisíveis por 3.

INDUÇÃO ESTRUTURAL

EXEMPLO: Seja o conjunto A definido como:

$$\begin{cases} 2 \in A \\ \text{se } x \in A \text{ e } y \in A\text{, então } x + y \in A \end{cases}$$

Mostre, por indução matemática, que os elementos de A são divisíveis por 2.

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

Caso base: somar 1 elemento (neste caso n=1)

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

Caso base: somar 1 elemento (neste caso n=1)

Chamadas Recursivas:

soma dos n elem = somar elemento n à soma dos n-1 elementos

soma dos n-1 elem = somar o elemento n-1 à soma dos n-2 elementos

soma dos n-2 elem = somar o elemento n-2 à soma dos n-3 elementos

....

soma dos 3 elem = somar o elemento 3 à soma dos 2 primeiros elementos

soma dos 2 elem = somar elemento 1 ao elemento 2

soma do 1 elem = somar 1 elemento (Caso base)

FONTE: Martins, C.

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

```
int soma_n(n) {
  if (n <= 1)
    return n;
  else
    return (n+ soma_n(n-1));
}
```

FONTE: Martins, C.

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

```
int soma_n(n) {
  if (n <= 1)
    return n;
  else
    return (n+ soma_n(n-1));
}
```

Caso base, condição em que facilmente se resolve o problema.

FONTE: Martins, C.

ALGORITMOS

EXEMPLO: Realizar a soma de n elementos

```
int soma_n(n) {
  if (n <= 1)
    return n;
  else
    return (n+ soma_n(n-1));
}
```

Chamadas recursivas, procurando simplificar o problema

FONTE: Martins, C.

RECURSÃO

EXERCÍCIO:

1) Dada a recursão (solucionada anteriormente) de $P(n) = a^n$, n natural, a inteiro, escreva o pseudo código para a solução do problema.

2) Encontre a solução fechada para a relação de recorrência, utilizando recursão

$T(1) = 1$
$T(n) = T(n-1) + 3 \quad , \quad n>1$

3) Considere a situação: Um colônia de morcegos é contada a cada dois meses. As quatro primeiras contagens foram 1200, 1800, 2700, 4050. Se essa taxa de crescimento continuar, qual será a 12ª contagem?

a) determine a relação de recorrência e expanda a recursão
b) defina, se possível, a solução fechada

RECURSÃO OU RECURSIVIDADE MATEMÁTICA - ALGORITMOS

EXERCÍCIO:

4) Implementar o código do cálculo de n! utilizando recursividade
*exemplo teste: 5!

5) Implementar a sequência de Fibonacci recursiva

6) Imprimir recursivamente os números naturais de 1 a n
 * exemplo teste n=10

7) Implemente um método recursivo que receba como entrada um número inteiro positivo n e retorne 1 + 2 + 3 + 4 + … + n

Referências - extras

https://ic.unicamp.br/~mc102/aulas/aula12.pdf

https://hanielbarbosa.com/teaching/ufmg/2020-1/ilc/notes/11-induction.pdf

https://homepages.dcc.ufmg.br/~camarao/ipcc/livro006.html

https://www.ime.usp.br/~pf/algoritmos/aulas/recu.html

https://panda.ime.usp.br/pensepy/static/pensepy/12-Recursao/recursionsimple-ptbr.html