### Recursividade

# Aula 06 Recursividade

Introdução ao Conceito

Programação II, 2020-2021

2021-04-14

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

### Recursividade

		. ~
67	Introd	lucão

Sumário

- 2 Definição
- 3 Complexidade
- 4 Relação de Recorrência
- 5 Exemplo 1: A Função Factorial
- 6 Relação de Recorrência: Síntese
- **7** Exemplo 2: Cálculo das Combinações
- 8 Relação de Recorrência: Classificação
- 9 Exemplo 3: Torres de Hanói
- Definição Recursiva: Condições de Terminação Casos Atípicos Casos com Interesse

Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Terminação Casos Atípicos

### Recursividade

		. ~
67	Introd	lucão

Sumário

- 2 Definição
- 3 Complexidade
- 4 Relação de Recorrência
- 5 Exemplo 1: A Função Factorial
- 6 Relação de Recorrência: Síntese
- **7** Exemplo 2: Cálculo das Combinações
- 8 Relação de Recorrência: Classificação
- 9 Exemplo 3: Torres de Hanói
- Definição Recursiva: Condições de Terminação Casos Atípicos Casos com Interesse

Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Terminação Casos Atípicos



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?
- Poderia dizer que é uma boneca oca que contém outra boneca oca, que contém outra e assim sucessivamente
- Mas também pode dar uma definição alternativa porventura mais simples:
  - cutra banaca *manyashka*.
- Este é um exemplo de uma definição recursiva.

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Hanói

Definição Recursiva:

Condições de

Terminação Casos Atípicos



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?

- Este é um exemplo de uma definição recursiva.

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?
- Poderia dizer que é uma boneca oca que contém outra boneca oca, que contém outra e assim sucessivamente.
- Este é um exemplo de uma definição recursiva.

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?
- Poderia dizer que é uma boneca oca que contém outra boneca oca, que contém outra e assim sucessivamente.
- Mas também pode dar uma definição alternativa, porventura mais simples:
  - Uma boneca matryoshka é uma boneca oca que contém outra boneca matryoshka.
- Este é um exemplo de uma definição recursiva.

### introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?
- Poderia dizer que é uma boneca oca que contém outra boneca oca, que contém outra e assim sucessivamente.
- Mas também pode dar uma definição alternativa, porventura mais simples:
  - Uma boneca matryoshka é uma boneca oca que contém outra boneca matryoshka.
- Este é um exemplo de uma definição recursiva.

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Terminação Casos Atípicos Casos com Interesse



- Se tivesse de descrever o que é uma boneca matryoshka, como o faria?
- Poderia dizer que é uma boneca oca que contém outra boneca oca, que contém outra e assim sucessivamente.
- Mas também pode dar uma definição alternativa, porventura mais simples:
  - Uma boneca matryoshka é uma boneca oca que contém outra boneca matryoshka.
- Este é um exemplo de uma definição recursiva.

### Introduc

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# Definição Recursiva

Uma definição de um conceito diz-se recursiva se envolve uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda nao entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904

### Recursividade

Introdução

. . .

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividad

Se ainda não entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904

### Introdução

### Complexidade

### Relação de

# Recorrência Exemplo 1: A Função

# Factorial

### Relação de Recorrência: Síntese

# Exemplo 2: Cálculo das Combinações

### Relação de Recorrência: Classificação

# Exemplo 3: Torres de

# Hanói Definicão Recursiva:

## Condições de Terminação

# **Definição Recursiva**

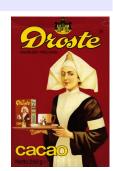
Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver recursividade.

Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Terminação Casos Atípicos

Casos com Interesse

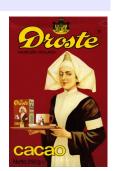
# Definição Recursiva

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver recursividade.

Podemos encontrar recursividade um pouco por todo o lado:



# Definição Recursiva

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904)

# Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação

# **Definição Recursiva**

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904)

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação

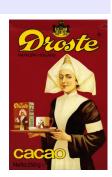
# **Definição Recursiva**

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904)

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação

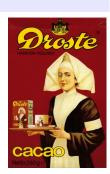
# **Definição Recursiva**

Uma definição de um conceito diz-se recursiva se envolver uma ou mais instâncias do próprio conceito.

### Recursividade

Se ainda não entendeu, ver *recursividade*. Podemos encontrar recursividade um pouco por todo o lado:

- Na descrição de certas formas geométricas.
- Nas imagens refletidas em espelhos paralelos.
- Na definição de certas funções matemáticas.
- Na sintaxe das linguagens de programação.



(circa 1904)

### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:
  - nos algoritmos;
  - nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 Como veremos, as definições recursivas podem também aparecer nos dois aspectos essenciais da programação:

- nos algoritmos;
- nas estruturas de dados.
- Usam-se definições recursivas porque frequentemente permitem descrever problemas complexos de forma simples.
- Desde Programação 1 temos vindo a apresentar e aplicar tecnologias e métodos para lidar com a complexidade inerente à resolução de problemas.
- Uma característica comum à maioria delas é o facto de reduzirem a redundância do código necessário para a solução.
- A estratégia consiste em tirar proveito das semelhanças formais entre as várias partes do código.

Introdução

Definição

Complexidad

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# Vejamos alguns casos

- Variáveis: permitem que o mesmo código produza resultados diferentes para dados diferentes.
- Instrução iterativa: permite substituir uma sequência de comandos estruturalmente semelhantes pela repetição de um único comando (geralmente parametrizado com variáveis auxiliares).
- Funções: permitem modularizar certas sequências de operações, transformando-as numa nova operação abstrata, que pode ser reutilizada múltiplas vezes.
   Promovem uma separação clara entre a utilização da função e a respectiva implementação. Quem a utiliza, delega a responsabilidade da resolução na função. Quem a implementa, pode livremente escolher o melhor algoritmo.

Introdução Definição

Complexidade

Relação de

Factorial

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Vejamos alguns casos:

- Variáveis: permitem que o mesmo código produza resultados diferentes para dados diferentes.
- Instrução iterativa: permite substituir uma sequência de comandos estruturalmente semelhantes pela repetição de um único comando (geralmente parametrizado com variáveis auxiliares).
- Funções: permitem modularizar certas sequências de operações, transformando-as numa nova operação abstrata, que pode ser reutilizada múltiplas vezes.
   Promovem uma separação clara entre a utilização da função e a respectiva implementação. Quem a utiliza, delega a responsabilidade da resolução na função. Quem a implementa, pode livremente escolher o melhor algoritmo.

Introdução

Definição

### Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Vejamos alguns casos:

- Variáveis: permitem que o mesmo código produza resultados diferentes para dados diferentes.
- Instrução iterativa: permite substituir uma sequência de comandos estruturalmente semelhantes pela repetição de um único comando (geralmente parametrizado com variáveis auxiliares).
- Funções: permitem modularizar certas sequências de operações, transformando-as numa nova operação abstrata, que pode ser reutilizada múltiplas vezes.
   Promovem uma separação clara entre a utilização da função e a respectiva implementação. Quem a utiliza, delega a responsabilidade da resolução na função. Quem a implementa, pode livremente escolher o melhor algoritmo.

Introdução

Definição Complexidade

Joinplexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Recorrência: Síntese Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atínicos

# Vejamos alguns casos:

- Variáveis: permitem que o mesmo código produza resultados diferentes para dados diferentes.
- Instrução iterativa: permite substituir uma sequência de comandos estruturalmente semelhantes pela repetição de um único comando (geralmente parametrizado com variáveis auxiliares).
- Funções: permitem modularizar certas sequências de operações, transformando-as numa nova operação abstrata, que pode ser reutilizada múltiplas vezes.
   Promovem uma separação clara entre a utilização da função e a respectiva implementação. Quem a utiliza, delega a responsabilidade da resolução na função. Quem a implementa, pode livremente escolher o melhor algoritmo.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atínicos

# Vejamos alguns casos:

- Variáveis: permitem que o mesmo código produza resultados diferentes para dados diferentes.
- Instrução iterativa: permite substituir uma sequência de comandos estruturalmente semelhantes pela repetição de um único comando (geralmente parametrizado com variáveis auxiliares).
- Funções: permitem modularizar certas sequências de operações, transformando-as numa nova operação abstrata, que pode ser reutilizada múltiplas vezes.
   Promovem uma separação clara entre a utilização da função e a respectiva implementação. Quem a utiliza, delega a responsabilidade da resolução na função. Quem a implementa, pode livremente escolher o melhor algoritmo.

Introdução

Definição

Complexidad

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# Relação de Recorrência

- O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?
- Se o problema se presta a ser descrito recursivamente então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

### Recursividade

Introdução

Definição

Complexidade

Relação de

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos Casos com Interesse

# Relação de Recorrência

- O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?
- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

# Introdução

Definição

Complexidade

### Relação Recorrê

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos Casos com Interesse

# Relação de Recorrência

 O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?

- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

Introdução

Definição

Complexidade

Relação Recorrêr

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atínicos

- O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?
- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

Definição

Complexidade

Relação d Recorrên

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?
- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

Definição

Complexidade

Relação o

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

função?

 O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria

- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

Introdução

Definição

Complexidade

Recorrên

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# Relação de Recorrência

 O caso das funções é particularmente interessante. Se quem as implementa é livre para escolher o melhor algoritmo, porque não escolher um que utiliza a própria função?

- Se o problema se presta a ser descrito recursivamente, então porque não implementá-lo da mesma forma?
- Para se poder fazer isso mesmo torna-se necessário ter uma descrição recursiva formal do problema: esse é o papel das Relações de Recorrência.
- Uma relação de recorrência é uma formulação recursiva formal de um problema.
- As relações de recorrência podem ser sempre implementadas de uma forma iterativa ou de uma forma recursiva.
- A implementação recursiva é estruturalmente muito próxima da própria relação de recorrência (donde resulta a sua simplicidade).

Introdução

Definição

Complexidade

Relação ( Recorrên

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Exemplo: a função factorial

Fórmula iterativa

$$n! = \begin{cases} \prod_{k=1}^{n} k, & n \in \mathbb{N} \\ 1, & n = 0 \end{cases}$$

Fórmula recursiva (relação de recorrência)

$$n! = \begin{cases} n \times (n-1)! & , n \in \mathbb{I} \\ 1 & , n = 0 \end{cases}$$

#### Recursividade

Introdução

Definição

Complexidade

Relação de

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula iterativa:

$$n! = \begin{cases} \prod_{k=1}^{n} k, & n \in \mathbb{N} \\ 1, & n = 0 \end{cases}$$

• Fórmula recursiva (relação de recorrência):

$$n! = \begin{cases} n \times (n-1)! & , n \in \mathbb{N} \\ 1 & , n = 0 \end{cases}$$

# Introdução

Definição

Complexidade Relação de

Recorrência

#### Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula iterativa:

$$n! = \begin{cases} \prod_{k=1}^{n} k, & n \in \mathbb{N} \\ 1, & n = 0 \end{cases}$$

Fórmula recursiva (relação de recorrência):

$$n! = \left\{ egin{array}{ll} n imes (n-1)! & , n \in \mathbb{N} \\ 1 & , n = 0 \end{array} \right.$$

Introdução

Definição

Complexidade

Relação de Recorrência

#### Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

```
static int factorial(int n)
{
   assert n >= 0;
   int result = 1;
   for (int i=2; i <= n; i++)
      result = result * i;
   return result;
}</pre>
```

$$n! = 1 \times 2 \times \cdots \times (n-1) \times r$$

O índice pode variar do caso limite C até ao valor *n*, ou *vice-versa*.

#### Implementação Recursiva

static int factorial(int n)
{
 assert n >= 0;
 int result = 1;
 if (n > 1)
 result = n \* factorial(n - 1);
 return result;

$$n! = n \times ((n-1) \times \cdots \times (2 \times (1)) \cdots$$

O argumento varia na direcção do caso limite (de n até 0)

#### ão de rrência

olo 1: A Função

ao de rrência: Síntese plo 2: Cálculo combinações

ião de rrência: ificação

plo 3: Torres de

ição Recursiva: ições de nação 3 Atípicos

s Atipicos s com Interesse

```
static int factorial (int n)
   assert n >= 0;
   int result = 1:
   for (int i=2; i <= n; i++)</pre>
      result = result * i:
   return result:
```

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n$$

O índice pode variar do caso limite 0 até ao valor n. ou vice-versa.

# rrência: Síntese ão de

$$n! = n \times ((n-1) \times \cdots \times (2 \times (1)) \cdots$$

1: A Função ão de

plo 2: Cálculo ombinações

rrência: ificação

plo 3: Torres de ição Recursiva:

icões de nação Atípicos

com Interesse

# Exemplo: a função factorial

#### Introdução Definição Complexidade

#### Implementação Iterativa

```
static int factorial(int n)
{
   assert n >= 0;
   int result = 1;
   for (int i=2; i <= n; i++)
      result = result * i;
   return result;
}</pre>
```

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n$$

O índice pode variar do caso limite 0 até ao valor *n*, ou *vice-versa*.

### Implementação Recursiva

```
static int factorial(int n)
{
   assert n >= 0;
   int result = 1;
   if (n > 1)
      result = n * factorial(n - 1);
   return result;
}
```

 $n! = n \times ((n-1) \times \cdots \times (2 \times (1)) \cdots$ 

O argumento varia na direcção do caso limite (de *n* até 0).

# ão de rrência

olo 1: A Função ial

ão de rrência: Síntese

combinações cão de rrência: ificação

plo 3: Torres de i

ição Recursiva: ições de nação s Atípicos

s Atipicos s com Interesse

```
static int factorial(int n)
{
   assert n >= 0;
   int result = 1;
   for (int i=2; i <= n; i++)
      result = result * i;
   return result;
}</pre>
```

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n$$

O índice pode variar do caso limite 0 até ao valor *n*, ou *vice-versa*.

# Implementação Recursiva

```
static int factorial(int n)
{
   assert n >= 0;
   int result = 1;
   if (n > 1)
      result = n * (factorial(n - 1);)
   return result;
}
```

 $n! = n \times ((n-1) \times \cdots \times (2 \times (1)) \cdots)$ O argumento varia na direcção do caso limite (de n até 0).

plo 1: A Funçã

ão de rrência: Síntese plo 2: Cálculo combinações

ao de rrência: ificação

ição Recursiva: ições de nação

Atípicos com Interesse

# Relação de Recorrência: Síntese

Método Iterativo (Repetitivo)

Método Recursivo

#### Recursividade

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

 O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

#### Método Recursivo

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Relação de Recorrência: Síntese

# Método Iterativo (Repetitivo)

 O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

#### Método Recursivo

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

 O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

#### Método Recursivo

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

#### Método Recursivo

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

 O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

#### Método Recursivo

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Método Iterativo (Repetitivo)
  - O algoritmo assenta num ciclo em que o índice pode variar desde o valor correspondente às situações limite até ao valor pretendido.

- Uma solução recursiva para um problema é expressa em função de si própria.
- Para que se atinja uma solução, cada invocação recursiva deve estar mais próxima de uma situação limite.
- Método poderoso e compacto de resolução de problemas mas potencialmente menos eficiente em termos de recursos pois tem de guardar o estado das várias invocações da função.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada
- Exemplo

$$C_{23}^{25} = rac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade Relação de

Recorrência
Exemplo 1: A Função

Factorial
Relação de

Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitariamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva:

Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva:

Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada.
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de

Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada.
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada.
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Fórmula:

$$C_k^n = \frac{A_k^n}{A_k^k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k!}$$
$$= \frac{n!}{(n-k)! \times k!}, \text{ com } n, k \in \mathbb{N}_0 \land n \ge k$$

- A aplicação destas fórmulas pode levantar problemas de cálculo numérico devido ao facto de os registos internos de armazenamento de um valor terem uma capacidade limitada.
- Exemplo:

$$C_{23}^{25} = \frac{15511210043330985984000000}{51704033477769953280000} = 300$$

- Para representar estes números necessitaríamos de pelo menos 84 bits (mesmo o tipo long tem apenas 64).
- Solução?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Exemplo 2: Combinações – Relação de Recorrência

Demonstração:

$$C_k^n = \frac{n!}{(n-k)! \times k!} = \frac{(n-1)! \times (k+n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)! \times k}{(n-k)! \times k!} + \frac{(n-1)! \times (n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)!}{(n-k)! \times (k-1)!} + \frac{(n-1)!}{(n-k-1)! \times k!}$$

$$= C_{k-1}^{n-1} + C_k^{n-1}$$

Relação de recorrência:

$$C_k^n=C_{k-1}^{n-1}+C_k^{n-1}$$
 , com  $n,k\in\mathbb{N}\wedge n>k$  (caso limite  $C_0^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite

#### Recursividade

Introdução

Definição Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

$$C_k^n = \frac{n!}{(n-k)! \times k!} = \frac{(n-1)! \times (k+n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)! \times k}{(n-k)! \times k!} + \frac{(n-1)! \times (n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)!}{(n-k)! \times (k-1)!} + \frac{(n-1)!}{(n-k-1)! \times k!}$$

$$= C_{k-1}^{n-1} + C_k^{n-1}$$

Relação de recorrência

 $C_k^n=C_{k-1}^{n-1}+C_k^{n-1}$  , com  $n,k\in\mathbb{N}\wedge n>k$   $C_0^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite  $C_0^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite

#### Recursividade

Introdução

Definição

Factorial

Complexidade Relação de

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Demonstração:

$$C_k^n = \frac{n!}{(n-k)! \times k!} = \frac{(n-1)! \times (k+n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)! \times k}{(n-k)! \times k!} + \frac{(n-1)! \times (n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)!}{(n-k)! \times (k-1)!} + \frac{(n-1)!}{(n-k-1)! \times k!}$$

$$= C_{k-1}^{n-1} + C_k^{n-1}$$

Relação de recorrência

 $C_k^n=C_{k-1}^{n-1}+C_k^{n-1}$  , com  $n,k\in\mathbb{N}\wedge n>k$   $C_0^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite  $C_n^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite

#### Recursividade

Introdução

Definição

Factorial

Complexidade

Relação de

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

$$C_k^n = \frac{n!}{(n-k)! \times k!} = \frac{(n-1)! \times (k+n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)! \times k}{(n-k)! \times k!} + \frac{(n-1)! \times (n-k)}{(n-k)! \times k!}$$

$$= \frac{(n-1)!}{(n-k)! \times (k-1)!} + \frac{(n-1)!}{(n-k-1)! \times k!}$$

$$= C_{k-1}^{n-1} + C_k^{n-1}$$

Relação de recorrência:

$$C_k^n=C_{k-1}^{n-1}+C_k^{n-1}$$
 , com  $n,k\in\mathbb{N}\wedge n>k$   $C_0^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite)  $C_n^n=1$  , com  $n\in\mathbb{N}_0$  (caso limite)

Introdução

Definição

Complexidade Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Terminação Casos Atípicos Casos com Interesse

```
- Simples;
- Compacto;
```

- Leyivei,

Fácil detectar erros

 E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

```
static int combNKK(int n, int k)
{
    assert 0 <= k && k <= n;
    int result = 1;
    if (k > 0 && k < n)
        result = combNKK(n-1, k-1) + combNKK(n-1, k);
    return result;
}</pre>
```

```
Simples;Compacto;Leafvel;
```

Fácil detectar erros.

 E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
      result = (combNKK(n-1, k-1)) + (combNKK(n-1, k);)
  return result;
}</pre>
```

```
Simples;Compacto;Larged
```

Fácil detectar erross

 E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
      result = (combNKK(n-1, k-1)) + (combNKK(n-1, k);)
  return result;
}</pre>
```

- Simples:
- Compacto;
- Legível;
- Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
      result = (combNKK(n-1, k-1)) + (combNKK(n-1, k);)
  return result;
}</pre>
```

- Simples;
- Compacto;
- Legível;
- Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Método Recursivo:
  - Simples;
  - Compacto;
  - Legível;
  - Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
      result = combNKK(n-1, k-1) + (combNKK(n-1, k);)
  return result;
}</pre>
```

- Método Recursivo:
  - Simples;
  - Compacto;
  - Legível;
  - Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
      result = (combNKK(n-1, k-1)) + (combNKK(n-1, k);)
  return result;
}</pre>
```

- Método Recursivo:
  - Simples;
  - Compacto;
  - Legível;
  - Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

```
static int combNKK(int n, int k)
{
  assert 0 <= k && k <= n;
  int result = 1;
  if (k > 0 && k < n)
    result = (combNKK(n-1, k-1)) + (combNKK(n-1, k);
  return result;
}</pre>
```

- Método Recursivo:
  - Simples;
  - Compacto;
  - Legível;
  - Fácil detectar erros.
- E se tentarmos implementar uma solução com o método iterativo?

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

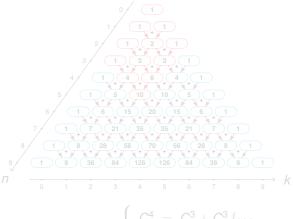
Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# Exemplo Combinações: Implementação Iterativa 1

#### Triângulo de Pascal:



$$C_2^5 = C_1^4 + C_2^4 \left\{ egin{array}{ll} C_1^4 = C_0^3 + C_1^3 \left\{ \cdot \cdot \right. \ C_2^4 = C_1^3 + C_2^3 \left\{ \cdot \cdot \right. \end{array} 
ight.$$

#### Recursividade

Introdução

Definição

Complexidade Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

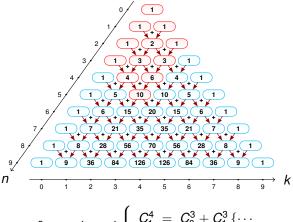
Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

• Triângulo de Pascal:



$$C_2^5 \ = \ C_1^4 + C_2^4 \left\{ egin{array}{ll} C_1^4 \ = \ C_0^3 + C_1^3 \left\{ \cdots 
ight. \ C_2^4 \ = \ C_1^3 + C_2^3 \left\{ \cdots 
ight. \end{array} 
ight.$$

#### Recursividade

Introdução

Definição

Factorial

Complexidade Relação de

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros)
- O processo iterativo pode seguir as regras seguintes

- O resultado é o elemento de índice k da linha n
- Este algoritmo pode ser optimizado considerando as seguintes factos:

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial Relação de

Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Casos com Interesse

 O programa mostrado a seguir faz todas essa ontimizações

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes
  - $\blacksquare$  existem n + 1 iterações (uma por linha):
  - a primeira linha (n=0) tem apenas o valor 1 (no posicão
    - k=0 do *array*), esse valor manter-se-á fixo para todas as linhas:
  - o para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i — 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando a seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sup>5</sup><sub>2</sub> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de

Recorrência: Classificação Exemplo 3: Torres de

Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - ② a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas;
  - (3) para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i – 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sub>2</sub><sup>5</sup> bastam os valores assinalados a vermelho na figura)
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função

Factorial Relação de

Recorrência: Síntese

das Combinações

Relação de

Recorrência: Classificação Exemplo 3: Torres de

Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - (3) para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i – 1 e i).
- O resultado é o elemento de índice k da linha n
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sup>5</sup><sub>2</sub> bastam os valores assinalados a vermelho na figura)
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de

Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas;
  - (3) para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sup>5</sup><sub>2</sub> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sup>2</sup> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade Relação de

Factorial

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sup>5</sup><sub>2</sub> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade

Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sub>2</sub><sup>5</sup> bastam os valores assinalados a vermelho na figura)
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade Relação de

Recorrência
Exemplo 1: A Função

Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sub>2</sub><sup>5</sup> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade Relação de

Factorial

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sub>2</sub><sup>5</sup> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Necessitamos de um array de k + 1 elementos para guardar os valores de uma linha (inicializado a zeros).
- O processo iterativo pode seguir as regras seguintes:
  - 1 existem n + 1 iterações (uma por linha);
  - 2 a primeira linha (n = 0) tem apenas o valor 1 (no posição k = 0 do array), esse valor manter-se-á fixo para todas as linhas:
  - 3 para as restantes n linhas, os valores do array desde o índice 1 até ao índice k são calculados como sendo a soma dos dois valores referidos pela relação de recorrência (se o índice do array for i, então será a soma dos valores com índice i - 1 e i).
- O resultado é o elemento de índice k da linha n.
- Este algoritmo pode ser optimizado considerando as seguintes factos:
  - Não é necessário calcular um triângulo completo (para C<sub>2</sub><sup>5</sup> bastam os valores assinalados a vermelho na figura).
  - O triângulo de Pascal é simétrico, por isso basta calcular metade.
- O programa mostrado a seguir faz todas essa optimizações.

Introdução Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

```
static int combIter1(int n, int k)
   assert 0 <= k && k <= n;
   int result = 1:
   if (k > 0 \&\& k < n) {
      int kMin = k < n-k ? k : n-k; // minimo(k, n-k)
      int[] linha = new int[k + 1];
      int c = 0;
      int cIni = 1:
      linha[0] = 1;
      for(int 1 = 1;1 <= n;1++) {
         if (1 > n-kMin+1)
            cIni++;
         for(c = kMin;c >= cIni;c--)
            linha[c] = linha[c]+linha[c-1];
      result = linha[kMin];
   return result:
```

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Há uma solução iterativa mais simples e mais eficiente
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha
- O algoritmo segue os passos:

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Casos Atípicos Casos com Interesse

 Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as sequintes

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha
- O algoritmo segue os passos:
  - Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte:
  - Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número n – k.
  - No fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade

Relação de

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é ε
    - Cada iteração do ciclo externo vai construir a diagonal sequinte:
    - Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número
  - No fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

#### Recursividade

Introdução

Definição

Complexidade

Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - ① Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - 2 Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - 4 O ciclo externo é repetido até chegar à diagonal número n – k
  - So No fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - 2 Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - 4 O ciclo externo é repetido até chegar à diagonal número n – k.
  - So fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade

Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número n – k.
  - So fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - 4 O ciclo externo é repetido até chegar à diagonal número n – k.
  - So fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade Relação de

**Factorial** 

Recorrência
Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número n – k.
  - No fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função

Factorial Relação de

Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número n – k.
  - No fim, o valor da posição k do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição

Complexidade Relação de

**Factorial** 

Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Há uma solução iterativa mais simples e mais eficiente.
- Baseia-se em construir os elementos necessários do triângulo diagonal-a-diagonal em vez de linha-a-linha.
- O algoritmo segue os passos:
  - 1 Começamos com um array de k + 1 elementos iniciados com uns, correspondendo aos elementos a vermelho na diagonal descendente mais à direita do triângulo. Esta é a diagonal zero.
  - Cada iteração do ciclo externo vai construir a diagonal seguinte;
  - 3 Para isso, o ciclo interno vai "descendo" ao longo da diagonal, adicionando a cada elemento do array (que ainda traz o valor da diagonal anterior) o elemento anterior do array (que tem o novo valor acabado de calcular).
  - O ciclo externo é repetido até chegar à diagonal número n – k.
  - **5** No fim, o valor da posição *k* do array tem o resultado pretendido.
- Na verdade, é mais simples iniciar o array com apenas um 1 na primeira posição e gerar a diagonal zero da mesma forma que as seguintes.

Introdução

Definição Complexidade

Relação de Recorrência

Exemplo 1: A Função

Relação de Recorrência: Síntese

Exemplo 2: Cálculo las Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

- Este algoritmo é preferível ao anterior porque percorre e gera apenas os elementos necessários, de forma regular e sem necessitar de condicões extra.
- O programa é mostrado a seguir

```
static int comblter2(int n,int k)
{
    assert 0 <= k && k <= n;
    int[] diag = new int[k+1];
    diag[0] = 1;
    for (int i = 0; i <= n-k; i++)
        for (int j = 1; j <= k; j++)
        diag[j] += diag[j-1];
    return diag[k];
}</pre>
```

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Hanói

Definição Recursiva:

Condições de Terminação Casos Atípicos

 Este algoritmo é preferível ao anterior porque percorre e gera apenas os elementos necessários, de forma regular e sem necessitar de condições extra.

O programa é mostrado a seguir.

```
static int combIter2(int n, int k)
{
    assert 0 <= k && k <= n;

    int[] diag = new int[k+1];
    diag[0] = 1;
    for (int i = 0; i <= n-k; i++)
        for (int j = 1; j <= k; j++)
            diag[j] += diag[j-1];
    return diag[k];
}</pre>
```

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação

- Este algoritmo é preferível ao anterior porque percorre e gera apenas os elementos necessários, de forma regular e sem necessitar de condições extra.
- O programa é mostrado a seguir.

```
static int combIter2(int n, int k)
{
    assert 0 <= k && k <= n;

    int[] diag = new int[k+1];
    diag[0] = 1;
    for (int i = 0; i <= n-k; i++)
        for (int j = 1; j <= k; j++)
            diag[j] += diag[j-1];
    return diag[k];
}</pre>
```

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Em termos de complexidade do mecanismo de descrição

- Simples: quando há apenas uma chamada recursiva
- Composta: quando há múltiplas chamadas recursivas

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

# Em termos de complexidade do mecanismo de descrição:

Simples: guando há apenas uma chamada recursivo

Composta: quando há múltiplas chamadas recursivas

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

### Em termos de complexidade do mecanismo de descrição:

- Simples: quando há apenas uma chamada recursiva.
  - Exemplo: factorial.
- Composta: quando há múltiplas chamadas recursivas.
  - Exemplo: combinações, torres de Hanói.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Em termos de complexidade do mecanismo de descrição:

- Simples: quando há apenas uma chamada recursiva.
  - Exemplo: factorial.
- Composta: quando há múltiplas chamadas recursivas.
  - Exemplo: combinações, torres de Hanói

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência:

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Em termos de complexidade do mecanismo de descrição:

- Simples: quando há apenas uma chamada recursiva.
  - Exemplo: factorial.
- Composta: quando há múltiplas chamadas recursivas.
  - Exemplo: combinações, torres de Hanói.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

Em termos de complexidade do mecanismo de descrição:

- Simples: quando há apenas uma chamada recursiva.
  - Exemplo: factorial.
- Composta: quando há múltiplas chamadas recursivas.
  - Exemplo: combinações, torres de Hanói.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

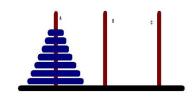
Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos

#### Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:

Sci pode illuver un disco de cada vez;
Mão pode colocar um disco em cima de outro de menor dimensão.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

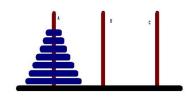
Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

#### Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:
  - Só pode mover um disco de cada vez
  - Não pode colocar um disco em cima de outro de menor dimensão.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

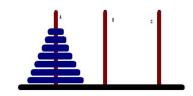
Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

#### Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:
  - Só pode mover um disco de cada vez;
  - Não pode colocar um disco em cima de outro de menor dimensão.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

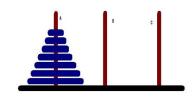
Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

# Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:
  - Só pode mover um disco de cada vez;
    - Não pode colocar um disco em cima de outro de menor dimensão.

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

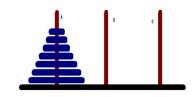
Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

# Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:
  - 1 Só pode mover um disco de cada vez;
    - Não pode colocar um disco em cima de outro de menor dimensão.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

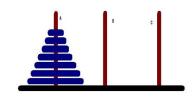
Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

# Exemplo 3: Torres de Hanói



- Este jogo, criado pelo matemático francês Édouard Lucas no Século XIX, é um dos exemplos clássicos que mostram as potencialidades dos algoritmos recursivos.
- Existem três postes onde se podem enfiar discos de diâmetros decrescente.
- O objectivo do jogo é mover todos os discos de um poste para outro, de acordo com as seguintes regras:
  - 1 Só pode mover um disco de cada vez;
  - 2 Não pode colocar um disco em cima de outro de menor dimensão.

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

## Torres de Hanói

# Relação de recorrência:

moverDiscos(n, tOrigem, tDestino, tAuxiliar

moverUmblisco (LOrigem , LDestino)

moverDiscostn-1, tAuxiliar, tDestino, tOrigeme

## Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar)
moverDiscos(tOrigem, tDestino)

# ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar
 moverDiscos(neces) fuedos

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

## Torres de Hanói

# Relação de recorrência:

moverDiscos(n, tOrigem, tDestino, tAuxiliar

moverUmDisco(tOr

#### Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar

## ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - 1 moverDiscos(n-1, tOrigem, tAuxiliar, tDestino
  - 2 moverUmDisco(tOrigem , tDestino
  - $\bigcirc$  moverDiscos(n-1, tAuxiliar, tDestino, tOrigem)

#### Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar)
moverDusciscos(tOrigem, tDestino)

# ou, alternativamente

• moverDiscos(0, tOrigem, tDestino, tAuxiliar (0.00 & process base reads)

Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - 1 moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - $\bigcirc$  moverDiscos(n-1, tAuxiliar, tDestino, tOrigem)

#### Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar)

## ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar)
 (não é proceso fezer nacia)

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - moverDiscos(n−1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos(n−1, tAuxiliar, tDestino, tOrigem

#### Caso limite

• moverDiscos(1, tOrigem, tDestino, tAuxiliar)

## ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar)
 (não é preciso fiscar nacia)

#### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - 1 moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos(n−1, tAuxiliar, tDestino, tOrigem)

#### Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar)

#### ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar)

#### Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - 1 moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos(n−1, tAuxiliar, tDestino, tOrigem)

#### Caso limite

moverDiscos(1, tOrigem, tDestino, tAuxiliar)

#### ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar)

#### Introdução

Definição

Complexidade Relação de

Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

## Torres de Hanói

# Relação de recorrência:

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - 1 moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos (n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

moverDiscos(1, tOrigem, tDestino, tAuxiliar)

#### ou, alternativamente

moverDiscos(0, tOrigem, tDestino, tAuxiliar)

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - ↑ moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos (n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

- moverDiscos(1, tOrigem, tDestino, tAuxiliar)
  - 1 moverUmDisco(tOrigem, tDestino)

## ou, alternativamente:

moverDiscos(0, tOrigem, tDestino, tAuxiliar)

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

#### Torres de Hanói

# Relação de recorrência:

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - ↑ moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos(n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

- moverDiscos(1, tOrigem, tDestino, tAuxiliar)
  - 1 moverUmDisco(tOrigem, tDestino)

## ou, alternativamente:

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

#### Torres de Hanói

# Relação de recorrência:

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - ↑ moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos(n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

- moverDiscos(1, tOrigem, tDestino, tAuxiliar)
  - 1 moverUmDisco(tOrigem, tDestino)

## ou, alternativamente:

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - ↑ moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos (n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

- moverDiscos(1, tOrigem, tDestino, tAuxiliar)
  - 1 moverUmDisco(tOrigem, tDestino)

## ou, alternativamente:

- moverDiscos(0, tOrigem, tDestino, tAuxiliar)
  - 1 (não é preciso fazer nada

#### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

- moverDiscos(n, tOrigem, tDestino, tAuxiliar)
  - ↑ moverDiscos(n-1, tOrigem, tAuxiliar, tDestino)
  - 2 moverUmDisco(tOrigem , tDestino)
  - 3 moverDiscos (n−1, tAuxiliar, tDestino, tOrigem)

## Caso limite:

- moverDiscos(1, tOrigem, tDestino, tAuxiliar)
  - 1 moverUmDisco(tOrigem, tDestino)

## ou, alternativamente:

- moverDiscos(0, tOrigem, tDestino, tAuxiliar)
  - 1 (não é preciso fazer nada)

#### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

#### Exemplo 3: Torres de

Definição Recursiva: Condições de Terminação Casos Atípicos

Introdução

Definição

Complexidade

```
static void moverDiscos(int n, String origem, String destino, String auxiliar)
{
   assert n >= 0;
   if (n > 0)
   {
      moverDiscos(n-1, origem, auxiliar, destino);
      out.println("Move disco "+n+" da torre "+origem+" para a torre "+destino);
      moverDiscos(n-1, auxiliar, destino, origem);
   }
}
```

- E se tentarmos implementar uma solução com o método iterativo?
- Existe solução para esse problema (como para qualquer outro algoritmo recursivo) mas a implementação é bastante complexa!

#### Hanói

Definição Recursiva: Condições de Terminação Casos Atípicos Casos com Interesse

Introdução

Definição

Complexidade

```
static void moverDiscos(int n, String origem, String destino, String auxiliar)
{
   assert n >= 0;
   if (n > 0)
   {
      moverDiscos(n-1, origem, auxiliar, destino);
      out.println("Move disco "+n+" da torre "+origem+" para a torre "+destino);
      moverDiscos(n-1, auxiliar, destino, origem);
   }
}
```

- E se tentarmos implementar uma solução com o método iterativo?
- Existe solução para esse problema (como para qualquer outro algoritmo recursivo) mas a implementação é bastante complexa!

#### Exemplo 3: Torres de

Introdução

Definição

Complexidade

```
static void moverDiscos(int n, String origem, String destino, String auxiliar)
{
    assert n >= 0;
    if (n > 0)
    {
        moverDiscos(n-1, origem, auxiliar, destino);
        out.println("Move disco "+n+" da torre "+origem+" para a torre "+destino);
        moverDiscos(n-1, auxiliar, destino, origem);
    }
}
```

- E se tentarmos implementar uma solução com o método iterativo?
- Existe solução para esse problema (como para qualquer outro algoritmo recursivo) mas a implementação é bastante complexa!

Exemplo 3: Torres de

Introdução

Definição

Complexidade

```
static void moverDiscos(int n, String origem, String destino, String auxiliar)
{
    assert n >= 0;
    if (n > 0)
    {
        moverDiscos(n-1, origem, auxiliar, destino);
        out.println("Move disco "+n+" da torre "+origem+" para a torre "+destino);
        moverDiscos(n-1, auxiliar, destino, origem);
    }
}
```

- E se tentarmos implementar uma solução com o método iterativo?
- Existe solução para esse problema (como para qualquer outro algoritmo recursivo) mas a implementação é bastante complexa!

#### Exemplo 3: Torres de

Introdução

Definição

Complexidade

Relação de Recorrência

Recursividade

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Terminação Casos Atípicos

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações Relação de

Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Para que uma função recursiva termine é preciso que:
  - Exista pelo menos uma alternativa n\u00e3o recursiva (CASO(S) LIMITE);
  - Todas as alternativas recursivas ocorram num contexto diferente do original (VARIABILIDADE);
  - 3 Em cada alternativa recursiva, o contexto (2) varie de forma a aproximar-se de um caso limite (1) (CONVERGÊNCIA).
- As condições (1) e (2) são necessárias. As três juntas são suficientes para garantir a terminação da recursão.

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Para que uma função recursiva termine é preciso que:
  - Exista pelo menos uma alternativa n\u00e3o recursiva (CASO(S) LIMITE);
  - Todas as alternativas recursivas ocorram num contexto diferente do original (VARIABILIDADE);
  - (3) Em cada alternativa recursiva, o contexto (2) varie de forma a aproximar-se de um caso limite (1) (CONVERGÊNCIA).
- As condições (1) e (2) são necessárias. As três juntas são suficientes para garantir a terminação da recursão.

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Casos Atípicos

- Para que uma função recursiva termine é preciso que:
  - 1 Exista pelo menos uma alternativa não recursiva (CASO(S) LIMITE);
  - Todas as alternativas recursivas ocorram num contexto diferente do original (VARIABILIDADE);
- As condições (1) e (2) são necessárias. As três juntas são

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de

Definição Recursiva: Condições de

- Para que uma função recursiva termine é preciso que:
  - Exista pelo menos uma alternativa n\u00e3o recursiva (CASO(S) LIMITE);
  - Todas as alternativas recursivas ocorram num contexto diferente do original (VARIABILIDADE);
  - 3 Em cada alternativa recursiva, o contexto (2) varie de forma a aproximar-se de um caso limite (1) (CONVERGÊNCIA).
- As condições (1) e (2) são necessárias. As três juntas são suficientes para garantir a terminação da recursão.

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese Exemplo 2: Cálculo

das Combinações Relação de

Recorrência: Classificação

Exemplo 3: Torres de Hanói

efinicão Recursiva:

Casos Atípicos

- Para que uma função recursiva termine é preciso que:
  - 1 Exista pelo menos uma alternativa não recursiva (CASO(S) LIMITE);
  - Todas as alternativas recursivas ocorram num contexto diferente do original (VARIABILIDADE);
  - 3 Em cada alternativa recursiva, o contexto (2) varie de forma a aproximar-se de um caso limite (1) (CONVERGÊNCIA).
- As condições (1) e (2) são necessárias. As três juntas são suficientes para garantir a terminação da recursão.

# **Análise dos Exemplos Apresentados**

Todos os exemplos de recursividade apresentados até agora verificam estas três condições:

- Factorial:
  - f(0) é um caso limite.
  - f(n) expresso em função de função
- Combinações
  - $\mathbb{C}(n,0)$  e G(n,n) são casos limite
  - C(n-1)(k-1)
- in converge para κ ou κ converge para U.
- Torres de Hanói
  - Mover 1 disco (ou 0 discos) é trivial.
  - move forre(n,...) expresso em função de sem função de s
  - moverone(n-1,...)
  - on converge para i (oull).

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

# **Análise dos Exemplos Apresentados**

Todos os exemplos de recursividade apresentados até agora verificam estas três condições:

• Factorial:

Combinações

Torres de Hanó

Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

## Factorial:

- f(0) é um caso limite
- 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$
- 3 A sucessão  $n, n-1, \ldots$  converge para 0.

# Combinações:

- $(n,0) \in C(n,n)$  são casos limite.
- $\bigcirc$  C(n, k) expresso em função de C(n-1, k) e
- n converge para k ou k converge para 0.

## Torres de Hanói:

- 1 Mover 1 disco (ou 0 discos) é trivial
- moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
- a n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n \neq n = 1$
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0
- Combinações:
  - $(n,0) \in C(n,n)$  são casos limite.
  - $\bigcirc$  C(n, k) expresso em função de C(n-1, k) e
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - ① Mover 1 disco (ou 0 discos) é trivial
  - moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - (a) n converge para 1 (ou 0)

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0
- Combinações:
  - $\bigcirc$  C(n,0) e C(n,n) são casos limite
  - $\bigcirc$  C(n, k) expresso em função de C(n-1, k) e C(n-1, k-1).
  - n converge para k ou k converge para 0.
- Torres de Hanói:
  - Mover 1 disco (ou 0 discos) é trivial
  - moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - a converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - ① C(n,0) e C(n,n) são casos limite
  - ② C(n, k) expresso em função de C(n-1, k) e
  - n converge para k ou k converge para 0
- Torres de Hanói:
  - Mover 1 disco (ou 0 discos) é trivial
  - moveTorre(n,...) expresso em função de moveTorre(n – 1,...).
  - a converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência:

Classificação Exemplo 3: Torres de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - ① C(n,0) e C(n,n) são casos limite.
    - 2 C(n, k) expresso em função de C(n-1, k) e C(n-1, k-1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 🕕 Mover 1 disco (ou 0 discos) é trivial
  - moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - 3 n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n-1, k) e C(n-1, k-1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 🕕 Mover 1 disco (ou 0 discos) é trivial
  - moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - (3) n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - n converge para k ou k converge para 0.
- Torres de Hanói:
  - 🕕 Mover 1 disco (ou 0 discos) é trivial
  - move lorre(n,...) expresso em função de moveTorre(n - 1,...).
  - n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 🕕 Mover 1 disco (ou 0 discos) é trivial
  - *move Torre*(n,...) expresso em função de *move Torre*(n-1,...).
  - (3) n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 1 Mover 1 disco (ou 0 discos) é trivial.
  - 2 moveTorre(n,...) expresso em função de moveTorre(n-1,...).
  - 3 n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 1 Mover 1 disco (ou 0 discos) é trivial.
  - ② moveTorre(n,...) expresso em função de moveTorre(n − 1,...).
  - 3 n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 1 Mover 1 disco (ou 0 discos) é trivial.
  - 2 moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - 3 n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

- Factorial:
  - f(0) é um caso limite.
  - 2 f(n) expresso em função de f(n-1) e  $n \neq n-1, \forall n$ .
  - 3 A sucessão  $n, n-1, \ldots$  converge para 0.
- Combinações:
  - 1 C(n,0) e C(n,n) são casos limite.
  - 2 C(n, k) expresso em função de C(n 1, k) e C(n 1, k 1).
  - 3 n converge para k ou k converge para 0.
- Torres de Hanói:
  - 1 Mover 1 disco (ou 0 discos) é trivial.
  - 2 moveTorre(n,...) expresso em função de moveTorre(n - 1,...).
  - 3 n converge para 1 (ou 0).

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Função McCarthy 91:

```
static int mc_carthy91(int n) (
    assert n > 0;
    int result;
    if (n > 100)
        result = n - 10;
    else
        result = mc_carthy91(mc_carthy91(n + 11));
    return result;
}
```

Conjectura de Collatz (3n + 1):

```
static long collatz(long n) {
   assert n > 0;
   long result = n;
   if (n == 1)
      result = 1;
   else if (n % 2 == 0)
      result = collatz(n / 2);
   else
      result = collatz(3 * n + 1);
   return result;
}
```

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

#### Terminação Casos Atípicos

• Função McCarthy 91:

```
static int mc_carthy91(int n) {
   assert n > 0;
   int result;
   if (n > 100)
      result = n - 10;
   else
      result = mc_carthy91(mc_carthy91(n + 11));
   return result;
}
```

Sabe-se que termina, mas o tipo complexo de recursão dificulta a demonstração.

Conjectura de Collatz (3n + 1):

```
static long collatz(long n) {
   assert n > 0;
   long result = n;
   if (n == 1)
      result = 1;
   else if (n % 2 == 0)
      result = collatz(n / 2);
   else
      result = collatz(3 * n + 1);
   return result;
}
```

Acredita-se que termina sempre, mas ninguém o demonstrou!

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Casos Atípicos

• Função McCarthy 91:

```
static int mc_carthy91(int n) {
   assert n > 0;
   int result;
   if (n > 100)
      result = n - 10;
   else
      result = mc_carthy91(mc_carthy91(n + 11));
   return result;
}
```

- Sabe-se que termina, mas o tipo complexo de recursão dificulta a demonstração.
- Conjectura de Collatz (3n + 1):

```
static long collatz(long n) {
   assert n > 0;
   long result = n;
   if (n == 1)
      result = 1;
   else if (n % 2 == 0)
      result = collatz(n / 2);
   else
      result = collatz(3 * n + 1);
   return result;
}
```

Acredita-se que termina sempre, mas ninguém o demonstrou!

#### Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

#### Casos Atípicos

• Função McCarthy 91:

```
static int mc_carthy91(int n) {
   assert n > 0;
   int result;
   if (n > 100)
      result = n - 10;
   else
      result = mc_carthy91(mc_carthy91(n + 11));
   return result;
}
```

- Sabe-se que termina, mas o tipo complexo de recursão dificulta a demonstração.
- Conjectura de Collatz (3n + 1):

```
static long collatz(long n) {
   assert n > 0;
   long result = n;
   if (n == 1)
      result = 1;
   else if (n % 2 == 0)
      result = collatz(n / 2);
   else
      result = collatz(3 * n + 1);
   return result;
}
```

Acredita-se que termina sempre, mas ninguém o demonstrou!

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Casos Atípicos

• Função McCarthy 91:

```
static int mc_carthy91(int n) {
   assert n > 0;
   int result;
   if (n > 100)
      result = n - 10;
   else
      result = mc_carthy91(mc_carthy91(n + 11));
   return result;
}
```

- Sabe-se que termina, mas o tipo complexo de recursão dificulta a demonstração.
- Conjectura de Collatz (3n + 1):

```
static long collatz(long n) {
   assert n > 0;
   long result = n;
   if (n == 1)
      result = 1;
   else if (n % 2 == 0)
      result = collatz(n / 2);
   else
      result = collatz(3 * n + 1);
   return result;
}
```

Acredita-se que termina sempre, mas ninguém o demonstrou!

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Recorrência: Classificação

Relação de

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Casos Atípicos

#### Casos com Interesse

#### Recursividade

Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de Terminação

Casos Atípicos

 Na área da programação, os problemas recursivos considerados são sempre problemas em que as três condições de terminação estão bem identificadas e podem ser implementadas. Introdução

Definição

Complexidade

Relação de Recorrência

Exemplo 1: A Função Factorial

Relação de Recorrência: Síntese

Exemplo 2: Cálculo das Combinações

Relação de Recorrência: Classificação

Exemplo 3: Torres de Hanói

Definição Recursiva: Condições de

Terminação