
Módulo II - Linguagens de Programação

Tópico 3 - Lógica de Programação e Algoritmos

—

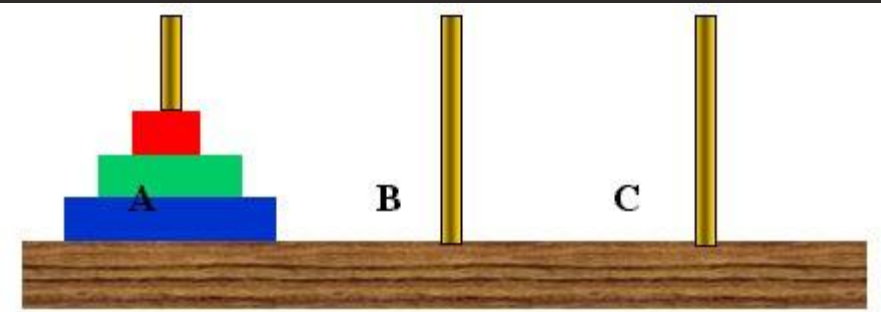
**Vamos começar com
um exercício mental!**

Esta é uma **Torre de Hanói**.

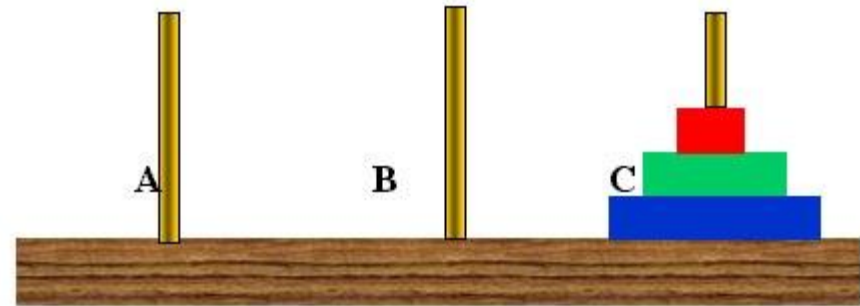
O objetivo é transferir os três anéis da haste A para C, usando B se necessário.

A regras são:

- deve-se mover um único anel por vez;
- um anel de diâmetro maior nunca pode repousar sobre um de diâmetro menor.



Situação inicial



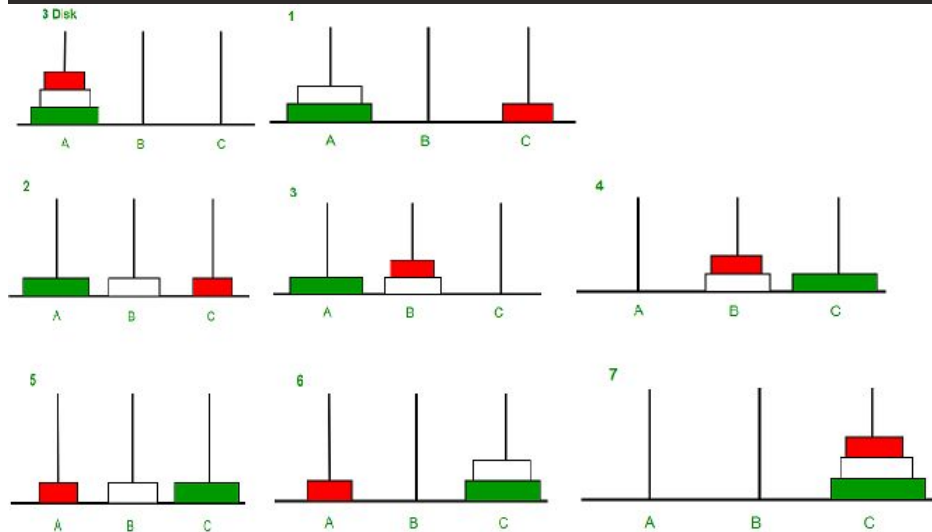
Situação final

Algoritmo para resolver o problema das Torres de Hanoi.

Início

1. Mover um anel da haste A para a haste C
2. Mover um anel da haste A para a haste B
3. Mover um anel da haste C para a haste B
4. Mover um anel da haste A para a haste C
5. Mover um anel da haste B para a haste A
6. Mover um anel da haste B para a haste C
7. Mover um anel da haste A para a haste C

Fim



—

Poderíamos resolver em **11**
passos, e também seria uma
resposta válida. Mas seria a
melhor solução?

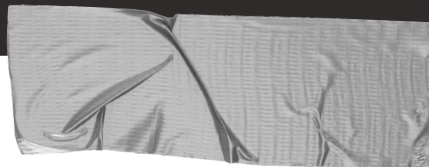
Algoritmo geral para resolver o problema das Torres de Hanoi.

Início

1. **Repita:**
2. Mova o menor anel de sua haste atual para a próxima no sentido anti-horário;
3. Execute o único movimento possível com um anel que não seja o menor de todos.
4. **Até** que todos os discos tenham sido transferidos para a outra haste.

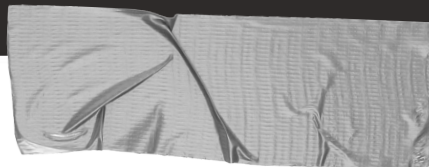
Fim





Revisão do conceito de algoritmo

1. Mat. Processo de cálculo, ou de resolução de um grupo de problemas semelhantes, em que se estipulam, com generalidade e sem restrições, regras formais para a obtenção do resultado, ou solução do problema.
2. Inform. Conjunto de regras e operações bem definidas e ordenadas destinadas à solução de um problema, ou de uma classe de problemas, em um número finito de etapas.



A formalização de um algoritmo

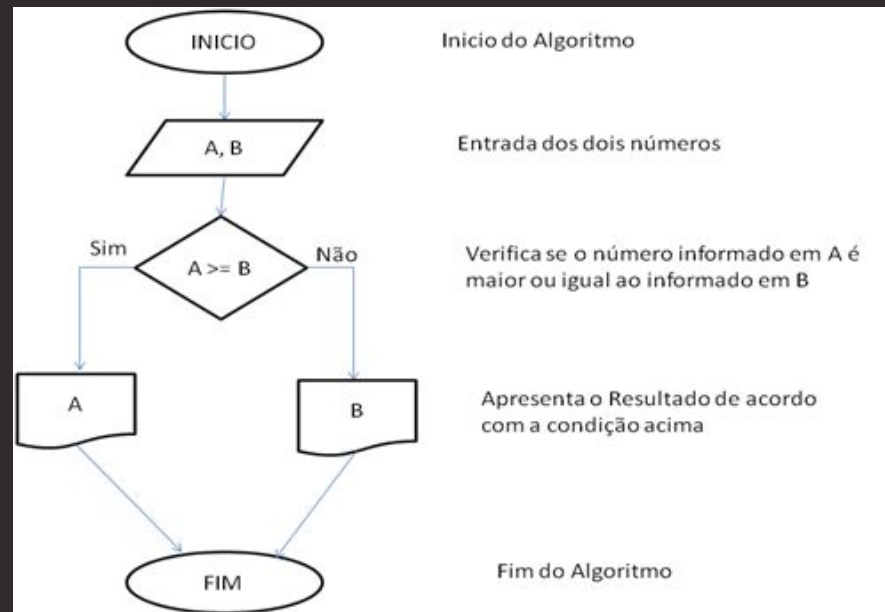
É importante que se siga alguma **convenção** na hora de escrever algoritmos, para que haja uma descrição formalizada e todas as pessoas envolvidas possam compreendê-lo da mesma forma.

Uma convenção tem definida um conjunto de regras de **sintaxe** para a escrita do algoritmo e também regras **semânticas**, que são as regras que permitem interpretar um algoritmo.








Início

1. **Ler** (A, B)
2. **Se** $A \geq B$ **Então**
3. **Exibir** (A)
4. **Senão**
5. **Exibir** (B)
6. **Fim Se**

Fim



Na formalização de um algoritmo temos bem definidas as palavras ou símbolos que utilizaremos. No caso de sua representação textual teremos **palavras reservadas**, como: ***Enquanto - Faça, Repita - Até, Se - Então, Ler(), Exibir(), Espere, Início, Fim.***

	Indica o início ou fim do processo
	Indica cada atividade que precisa ser executada
	Indica um ponto de tomada de decisão
	Indica a direção do fluxo
	Indica os documentos utilizados no processo
	Indica uma espera
	Indica que o fluxograma continua a partir desse ponto em outro círculo, com a mesma letra ou número, que aparece em seu interior

—

Qual o papel da **lógica** na
programação?

—

Lógica é uma área da Matemática cujo objetivo é investigar a veracidade de suas proposições.

—

1. **Se** estiver chovendo, levarei meu guarda-chuva
2. Está chovendo

—

1. **Se** estiver chovendo, levarei meu guarda-chuva
2. Está chovendo
3. **Então**, levarei meu guarda-chuva

—

1. **Se** estiver chovendo, levarei meu guarda-chuva
2. Eu peguei meu guarda-chuva.
3. **?**

-
1. **Se** estiver chovendo, levarei meu guarda-chuva
 2. Eu peguei meu guarda-chuva.
 3. **Então** é plausível que esteja chovendo.

Não posso afirmar com precisão que está chovendo só porque peguei meu guarda-chuva!

—

Há uma diferença de natureza entre esses argumentos. Enquanto em um eu posso dizer **necessariamente** o que se segue, o outro eu tenho alguma pista mas não posso dizer com precisão o que se segue.

—

Chamamos esses argumentos em que a conclusão se segue necessariamente de **argumentos dedutivos**, eles são **deduções**.

—

Chamamos esses outros tipos de argumentos em que a conclusão **não** se segue necessariamente de **argumentos indutivos**, eles são **induições**.

No contexto da computação só
vamos nos ocupar de **deduções!**