* Introdução aos algoritmos

- programa/software é representado pelas instruções e dados que algum humano definiu e que ao serem executados por alguma máquina cumprem algum objetivo.
- dado: é um valor qualquer armazenado em um computador
- informação: interpretação deste dado, ou seja, qual o seu significado.

01 Etapas de desenvolvimento de um programa

- Análise: criam-se especificações que detalham como o programa vai funcionar
- Projeto: criam-se especificações que detalham o resultado da análise em termos mais

próximos da implementação do programa

- **Implementação:** usando uma linguagem de programação e as especificações do projeto, o software é construído

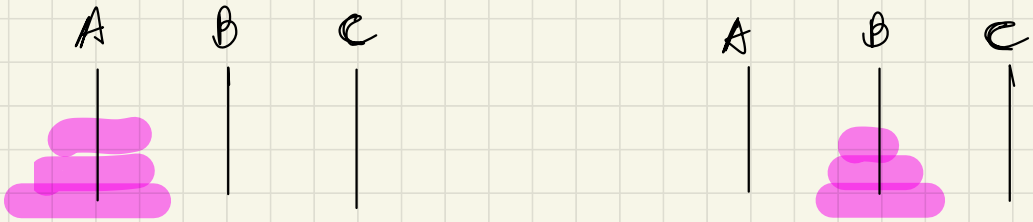
- **Testes:** são realizados testes para conferir sua conformidade com os requisitos iniciais

02 Algoritmos e Lógica de Programação

↳ **algoritmo:** Conjunto de regras para a solução de um problema. Ex: receita de bolo. Descreve as regras necessárias para a conclusão de um objetivo, além dos ingredientes e suas quantidades.

Obs: a correta execução das instruções leva à sua preparação.

EXEMPLO DE ALGORITMO: torres de Hanoi



situação inicial

situação final

OBJETIVO: mover os três discos da haste A para B, usando C se necessário. As regras de movimento são:

- deve-se mover um único disco por vez
- Um disco maior nunca pode repousar sobre algum disco menor

Possível Solução:

Algoritmo 1.1 Resolução Torres de Hanoi

Início

1. Mover um disco de A para B
2. Mover um disco de A para C
3. Mover um disco de B para C
4. Mover um disco de A para B
5. Mover um disco de C para A
6. Mover um disco de C para B
7. Mover um disco de A para B

Fim

TODO: Resolver em sala. Nomear os discos

03 Exercitando o Raciocínio

↳ o primeiro obstáculo entre você e seu primeiro programa de verdade é a habilidade de separar os problemas em pequenas ações executáveis que um computador possa fazer por você.

Ex: separar a atividade de pescar em um conjunto de instruções simples para que um robô pesque para você.

* 1ª tentativa: Algoritmo para Pescar

1. Coloque a minhoca no anzol
2. Arremesse a linha no lago/água
3. Observe a boia até que ela seja puxada para dentro da água

4. PUXE A LINHA e o peixe

5. Se ACABOU de PESCAR, VOLTE PARA CASA.
SE NÃO, VOLTE AO PASSO 1.

- seguir estas declarações produzirão um resultado (esperamos pegar o peixe)
- algumas instruções são simples (2, 4)
- outras dependem de alguma condição (a boia esta acima ou dentro da água?)
- instruções podem redirecionar o andamento da receita, como "se não acabou de pescar, então volte ao começo e coloque outra minhoca no anzol"
- Declarações ou Instruções são a base da computação / programação

* Computadores fazem EXATAMENTE o que dizemos a eles. Olhemos para nosso algoritmo da pesca: quais problemas podemos encontrar?

- [ ] Se não houver peixes, ficaremos pescando para sempre.
- [ ] Se a minhoca cair do anzol, nunca saberemos ou a substituiremos
- [ ] O que acontece se as minhocas acabarem
- [ ] Especificamos o que fazer com o peixe depois de pescá-lo?
- [ ] O que aconteceu com a vara de pescar?
- [ ] ... mais problemas?

* Quais os objetos são usados em nossa receita de pescar?

+ Problemas na pesca:

- [ ] Há especificação sobre o que é um bom arremesso? Se não cair na água?
- [ ] Normalmente, quando a boia afunda, você precisa fisgar o peixe primeiro.
- [ ] Como sabemos se terminamos de pescar? Pela hora? Quando acabarem as minhocas? Outra coisa?