Busca

Prof. Andrei Braga
Prof. Geomar Schreiner

Busca

- Uma busca consiste em recuperar um ou mais itens armazenados em um repositório de dados.

- Sempre buscamos os dados da mesma forma?

Busca

- Uma busca consiste em recuperar um ou mais itens armazenados em um repositório de dados.

- Sempre buscamos os dados da mesma forma?
    - Depende!

Busca

- Uma busca consiste em recuperar um ou mais itens armazenados em um repositório de dados.

- Sempre buscamos os dados da mesma forma?
    - Depende!

Busca

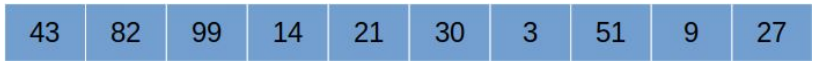
- Uma busca consiste em recuperar um ou mais itens armazenados em um repositório de dados.
- Depende:
    - De como os dados estão estruturados
    - Vetor, lista, árvore, arquivo
    - Se os dados estão ou não ordenados
    - Se há duplicidade de chaves

Busca Linear

- Método mais simples de pesquisa
- Varredura serial do conjunto de dados, da primeira até a última posição, comparando a chave de pesquisa com a chave de cada entrada
  - pesquisa bem-sucedida: é encontrada uma chave igual
  - pesquisa malsucedida: o final da lista é atingido sem que a chave procurada seja encontrada.
- Pode-se retornar:
  - o próprio elemento encontrado; ou
  - o índice do elemento (no caso de um vetor).

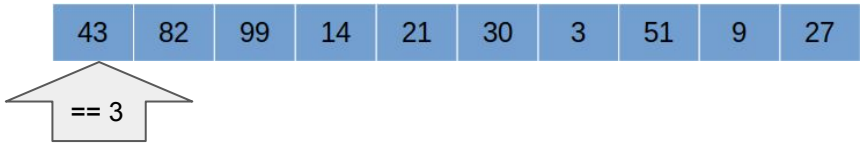
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



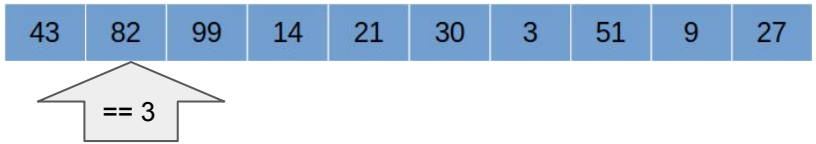
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



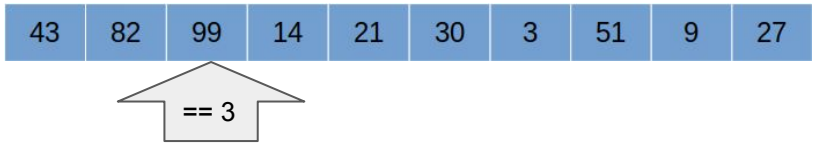
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



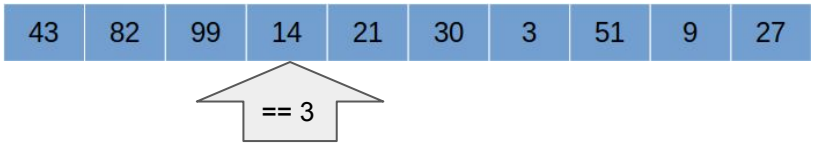
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



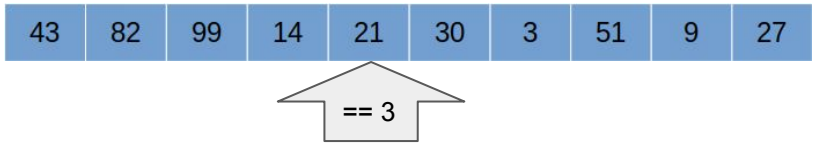
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



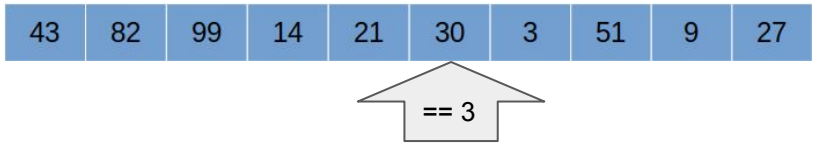
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



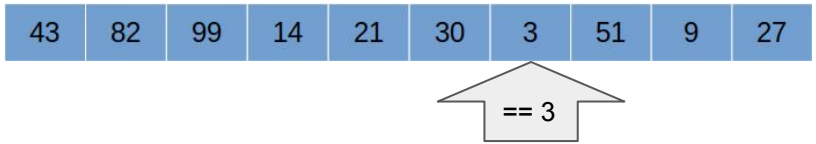
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



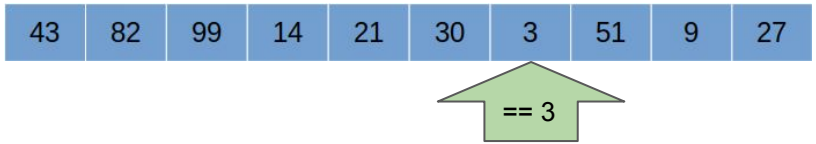
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



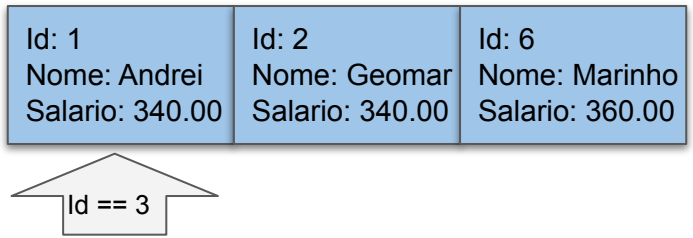
Busca Linear

- Vamos considerar primeiramente um vetor
  - Buscar a chave 3



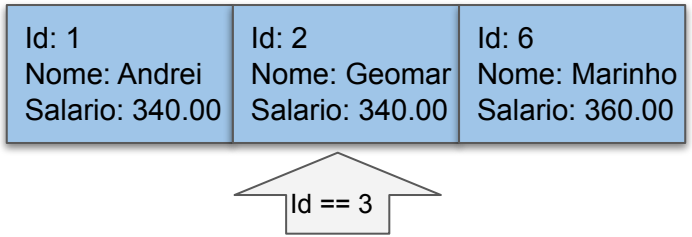
Busca Linear

- E se fosse um vetor de struct
  - Buscar Funcionário com id = 5



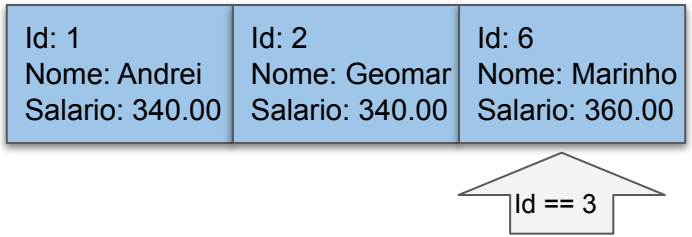
Busca Linear

- E se fosse um vetor de struct
  - Buscar Funcionário com id = 5



Busca Linear

- E se fosse um vetor de struct
  - Buscar Funcionário com id = 5



Busca Linear

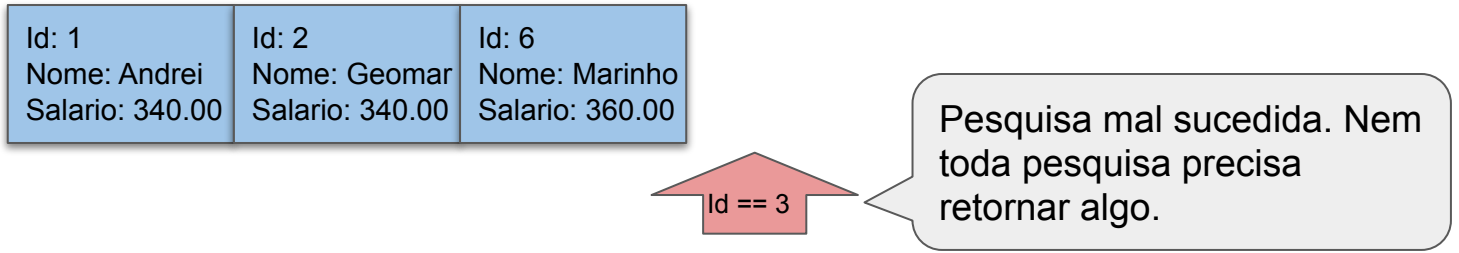
- Ese fosse um vetor de struct
  - Buscar Funcionário com id = 5

Id: 1 Nome: Andrei Salario: 340.00	Id: 2 Nome: Geomar Salario: 340.00	Id: 6 Nome: Marinho Salario: 360.00

Id == 3

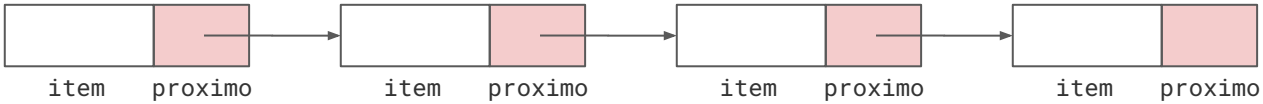
Busca Linear

- Ese fosse um vetor de struct
  - Buscar Funcionário com id = 5



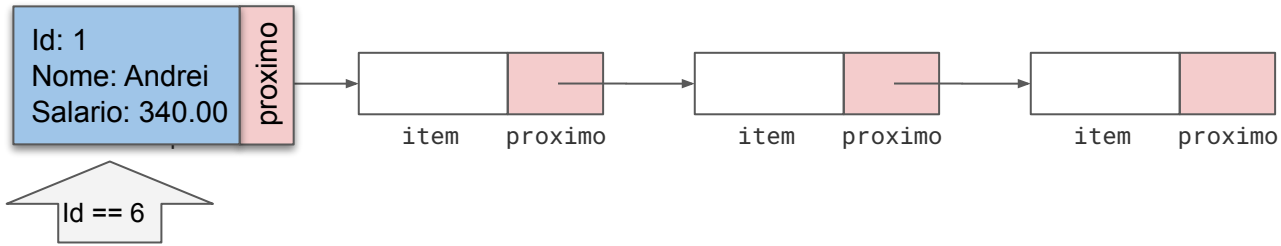
Busca Linear

- E se fosse uma lista encadeada
  - Buscar Funcionário com id = 6



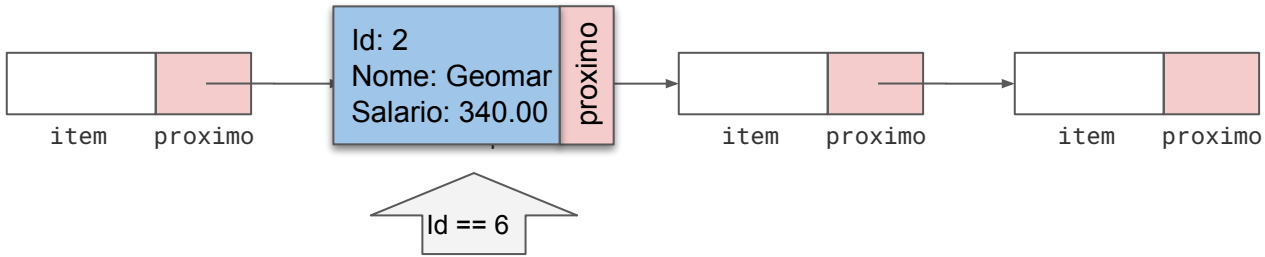
Busca Linear

- Ese fosse uma lista encadeada
  - Buscar Funcionário com id = 6



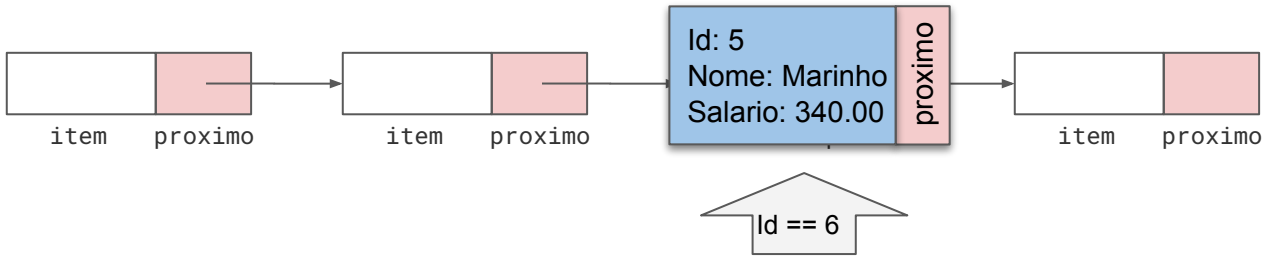
Busca Linear

- Ese fosse uma lista encadeada
  - Buscar Funcionário com id = 6



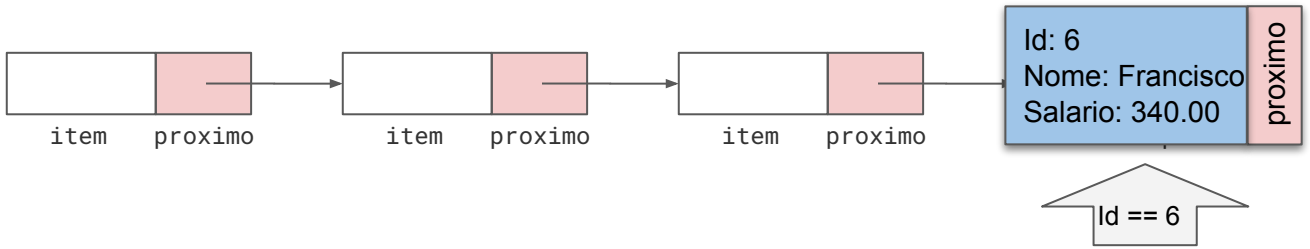
Busca Linear

- Ese fosse uma lista encadeada
  - Buscar Funcionário com id = 6



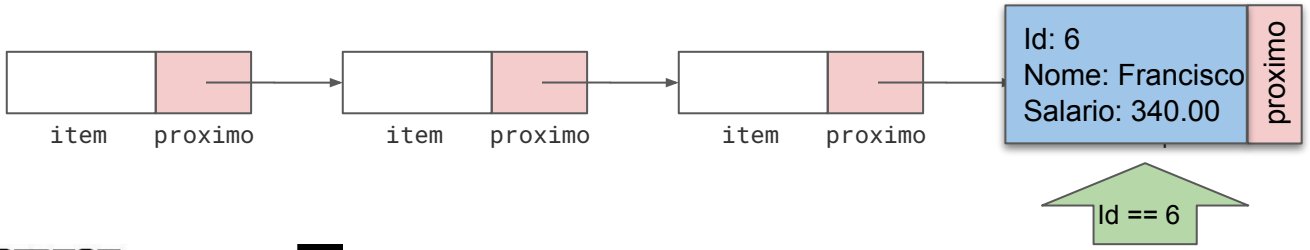
Busca Linear

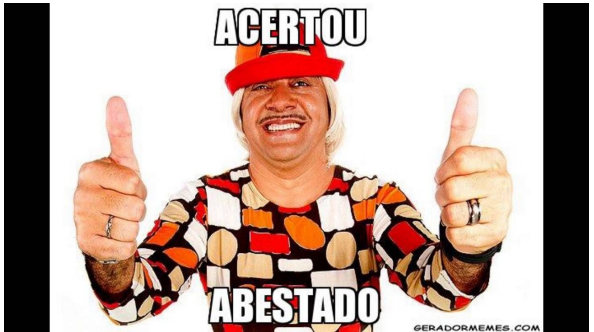
- Ese fosse uma lista encadeada
  - Buscar Funcionário com id = 6



Busca Linear

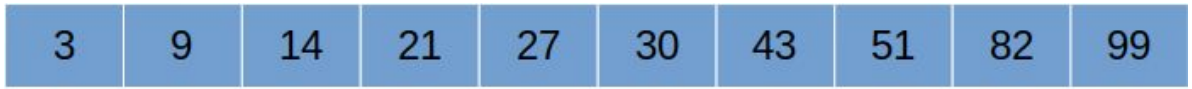
- Ese fosse uma lista encadeada
  - Buscar Funcionário com id = 6





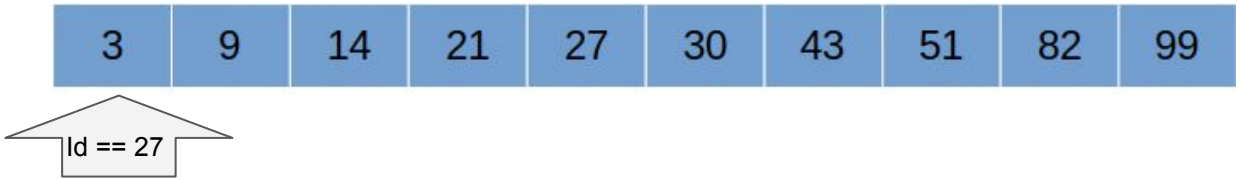
Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?



Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?



Busca Linear

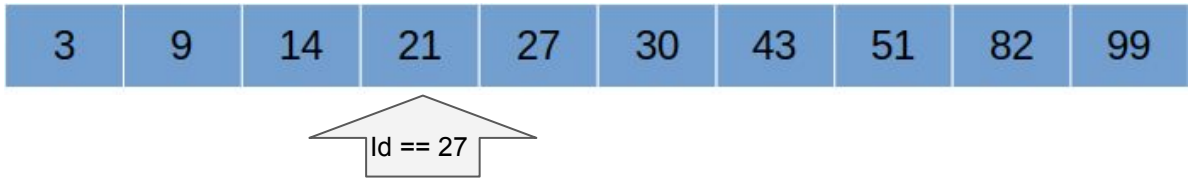
- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?

Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?

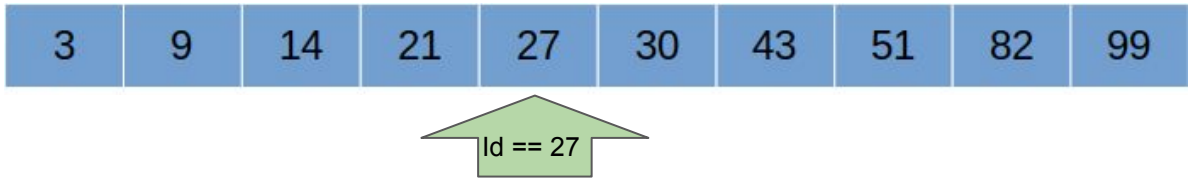
Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?



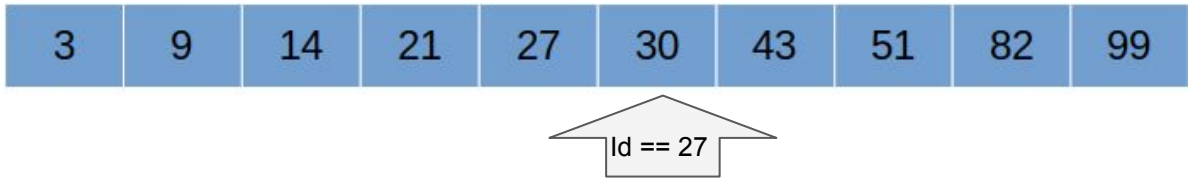
Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?



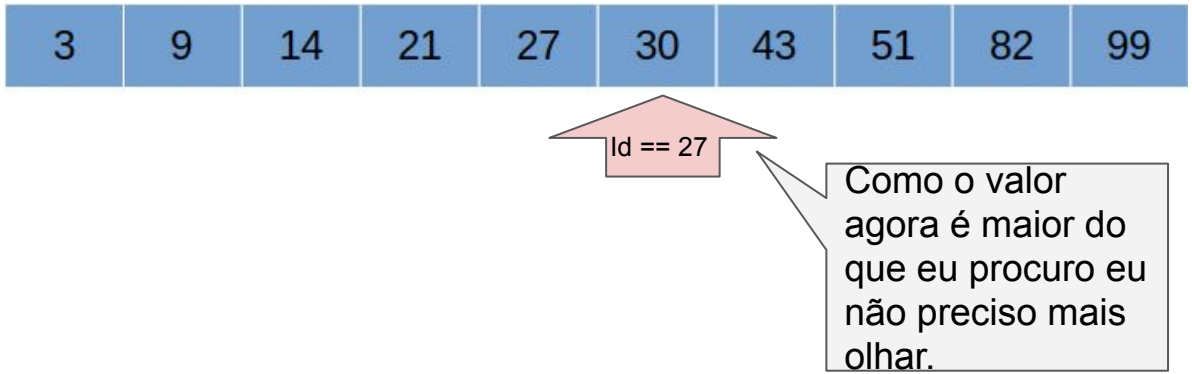
Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?



Busca Linear

- Esse método funciona para qualquer tipo de estrutura
  - Simples e robusto
  - Oneroso pois passa por todos os elementos
- Se os valores da estrutura estiverem ordenados podemos otimizar?

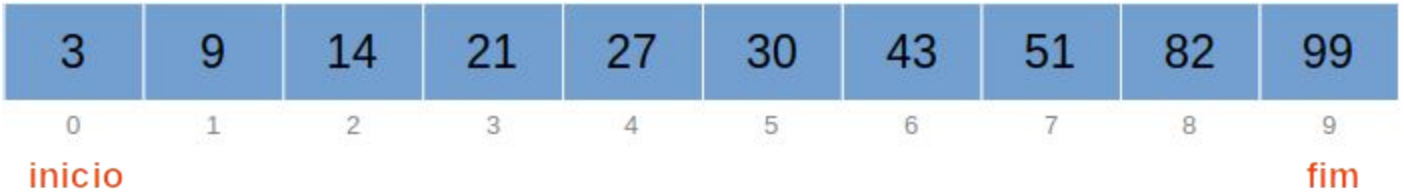


Busca Binária

- Considerando um vetor pré ordenado, podemos tirar vantagem para otimizar o código
- Num vetor ordenado, podemos adotar uma estratégia mais sofisticada e eficiente: busca binária
- Divisão e conquista: a cada passo, analisa o elemento do meio do vetor.
  - Caso 1. O elemento do meio corresponde à chave procurada
    - a busca termina com sucesso.
  - Caso 2. A chave buscada é menor do que o elemento do meio
    - a busca continua na primeira metade do vetor.
  - Caso 3: A chave buscada é maior do que o elemento do meio
    - a busca continua na segunda metade do vetor.

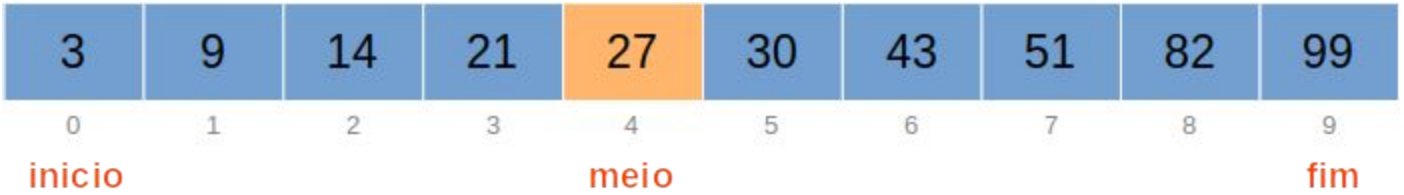
Busca Binária

- Buscar pelo elemento 43



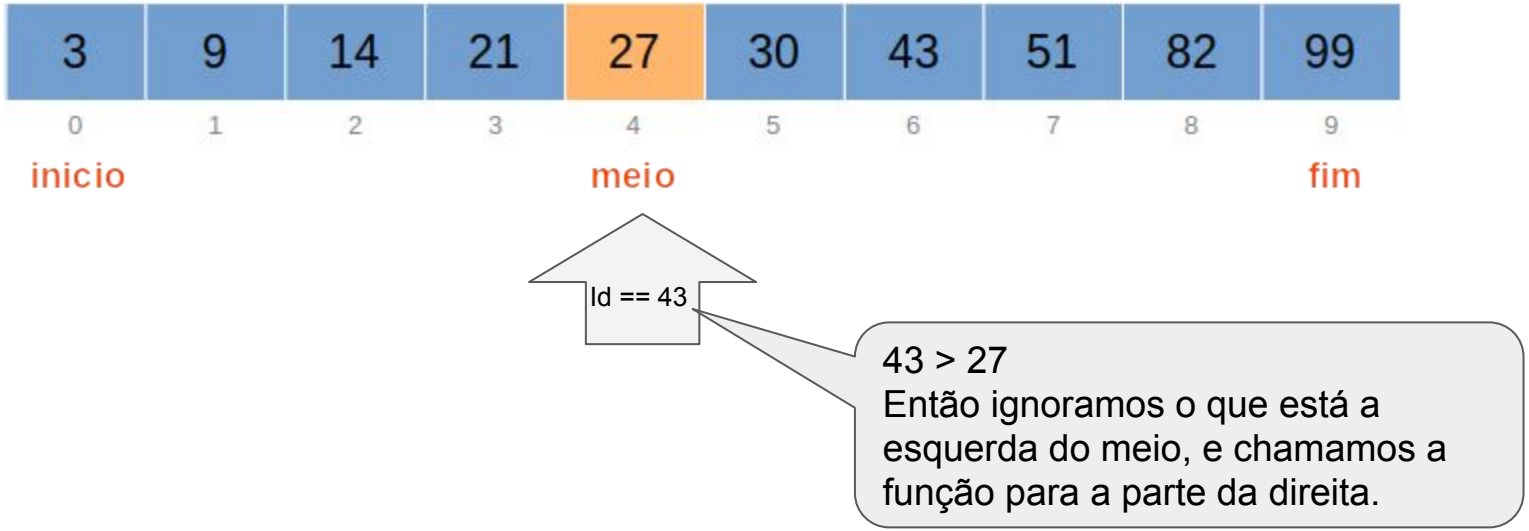
Busca Binária

- Buscar pelo elemento 43



Busca Binária

- Buscar pelo elemento 43

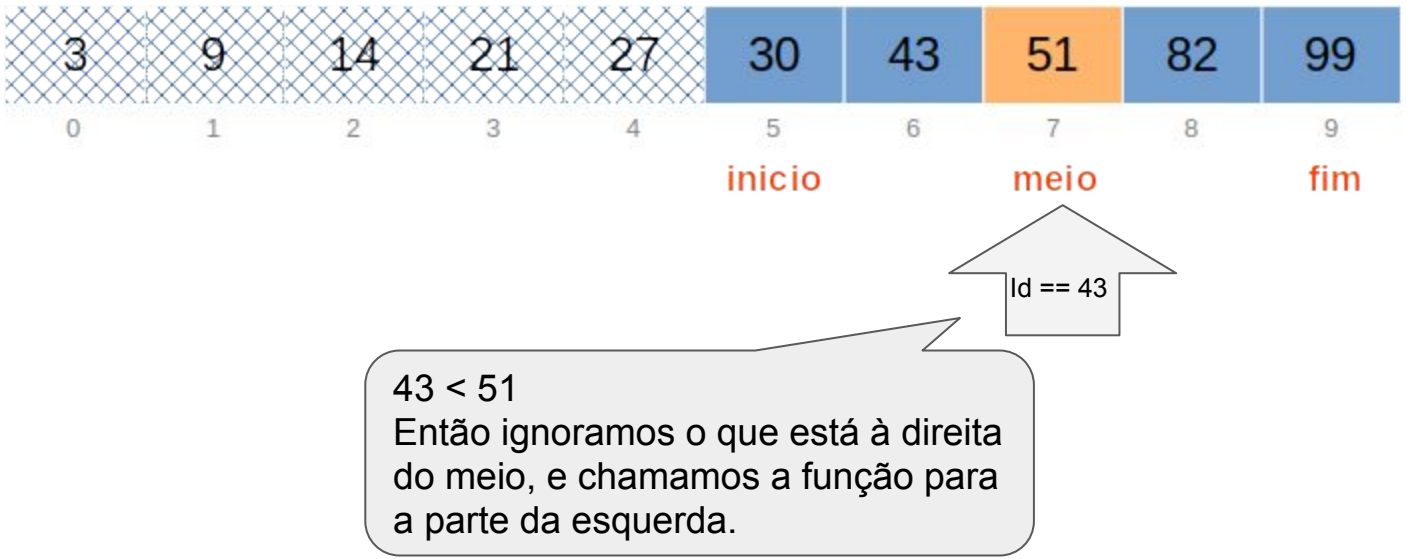


Busca Binária

- Buscar pelo elemento 43

Busca Binária

- Buscar pelo elemento 43

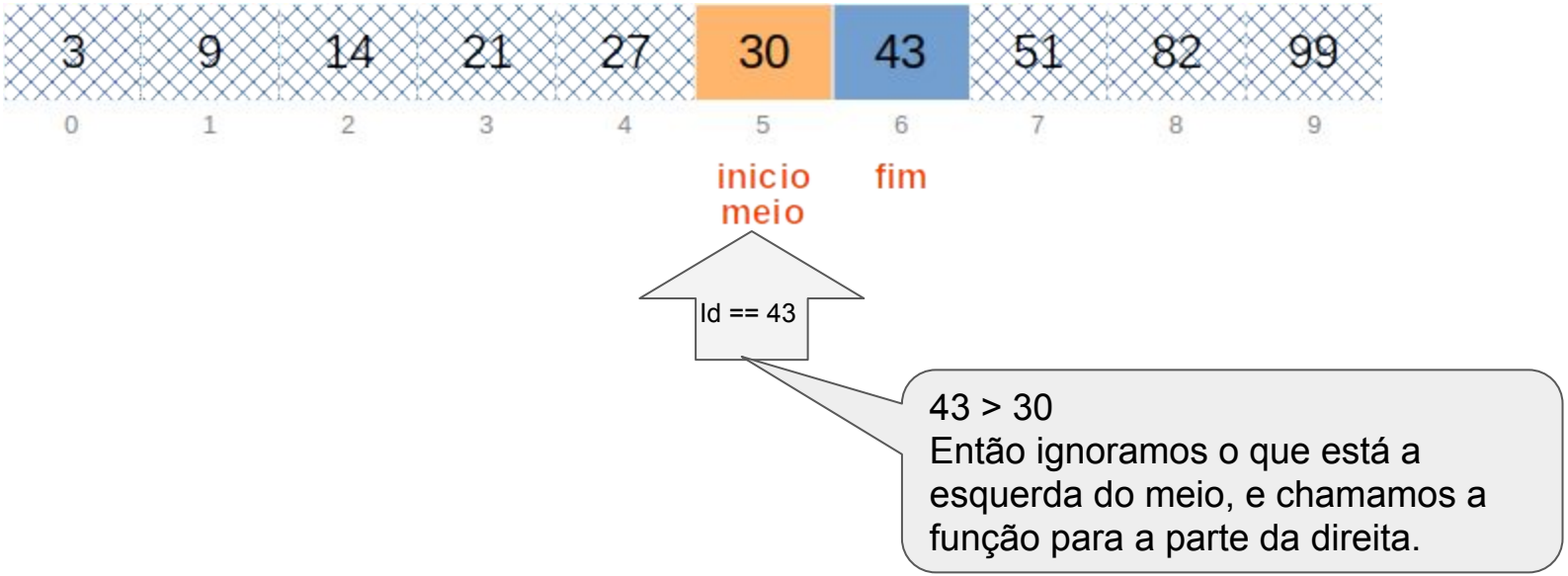


Busca Binária

- Buscar pelo elemento 43

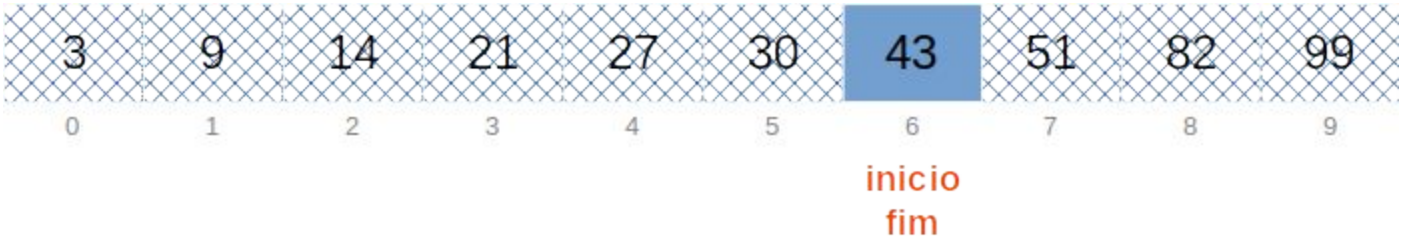
Busca Binária

- Buscar pelo elemento 43



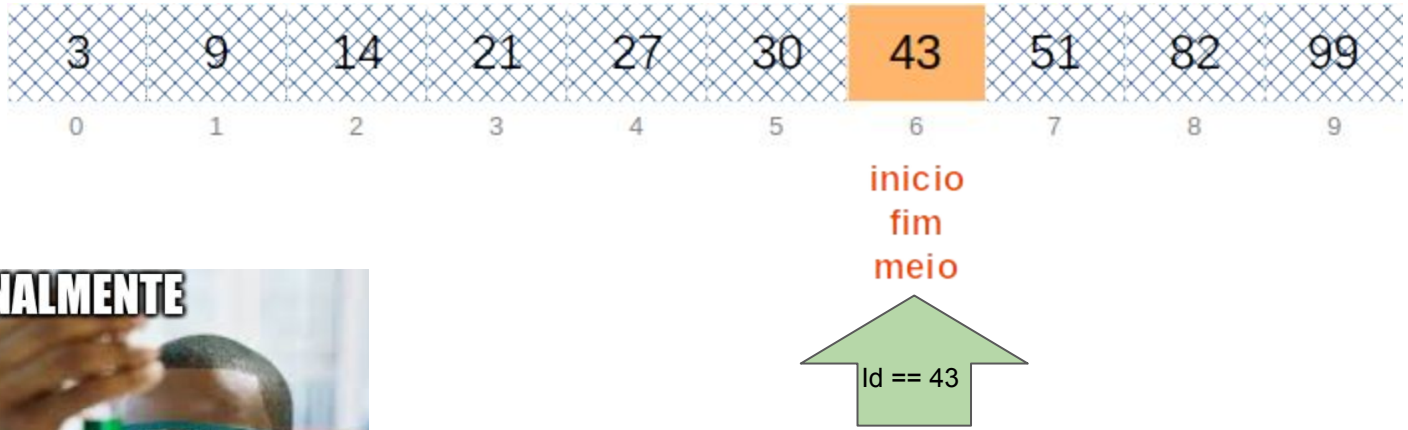
Busca Binária

- Buscar pelo elemento 43



Busca Binária

- Buscar pelo elemento 43



Busca Binária

- É um algoritmo relativamente simples
  - Utiliza o paradigma dividir para conquistar
- Possui um desempenho superior a busca linear
- Depende de um vetor ordenado para funcionar
  - O vetor pode ser de qualquer tipo
  - É possível implementá-lo em uma lista encadeada mas perde eficiência
- Implementação mais usual utiliza recursividade

Busca Binária

- Algoritmo

```
int buscaBinaria (int *vet, int inicio, int fim, int chave){
   int meio;
   if (inicio > fim)
       return -1;
   meio = (inicio+fim)/2;

   if (vet[meio] == chave) {
       return meio;
   }

   if (chave > vet[meio] )
       return buscaBinaria(vet, meio+1, fim, chave);
   else
       return buscaBinaria(vet, inicio, meio-1, chave);

}
```