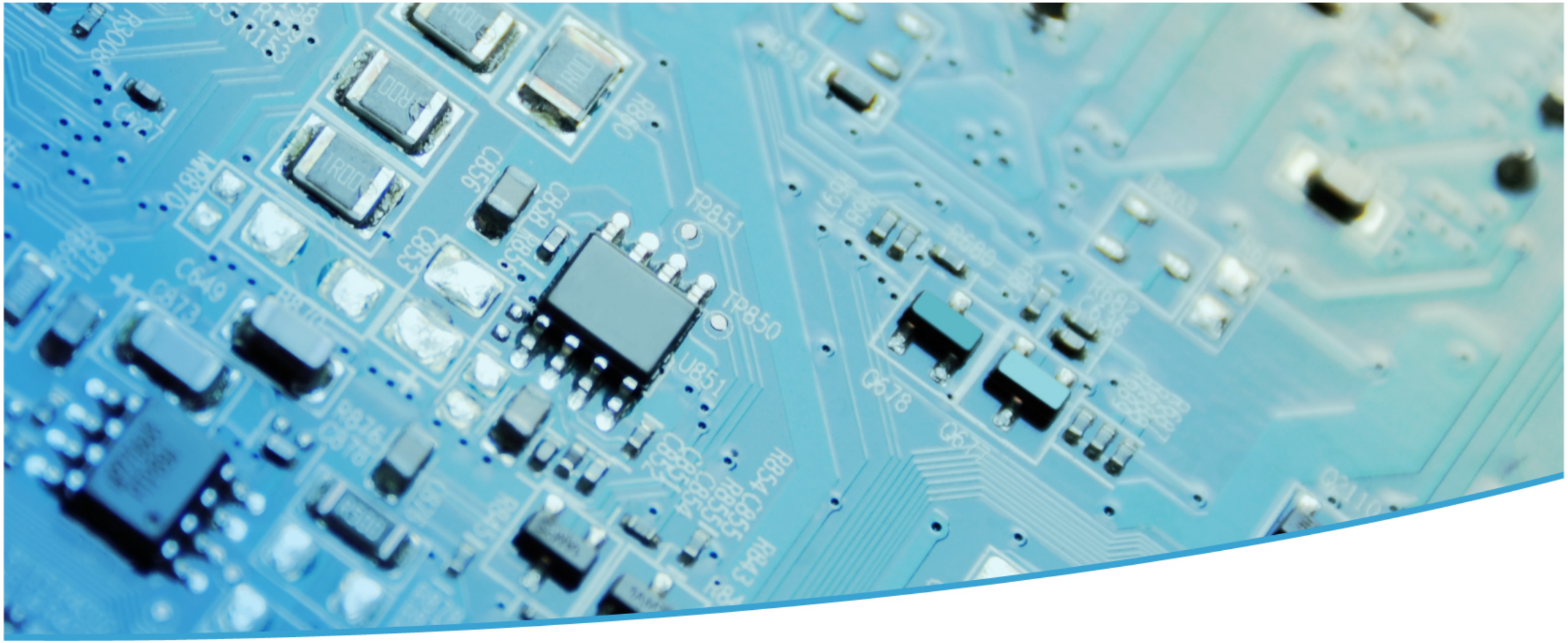
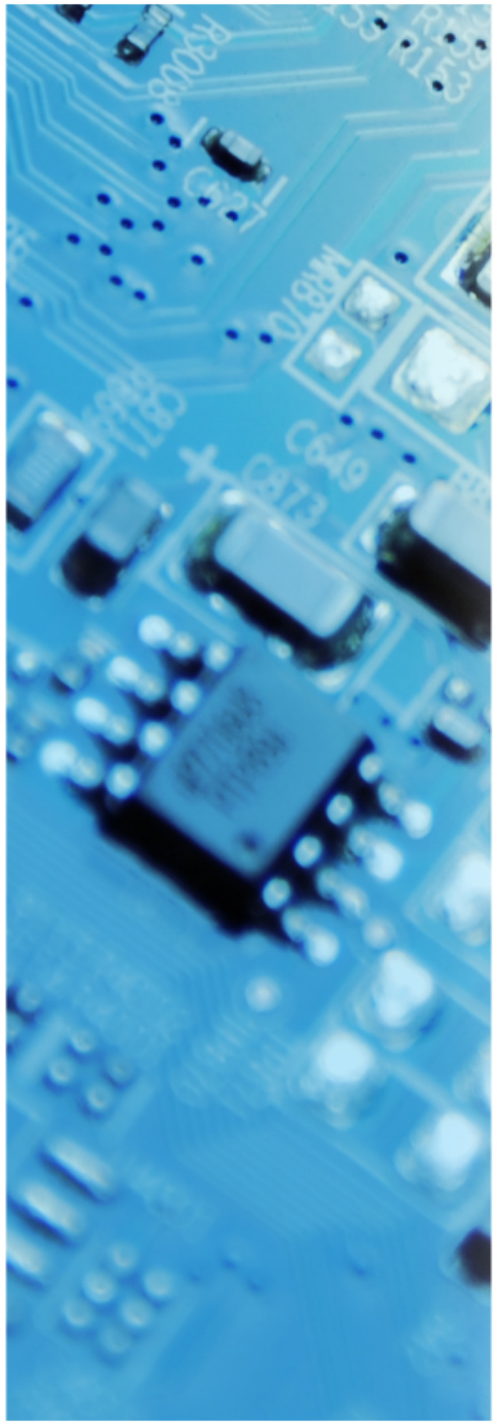
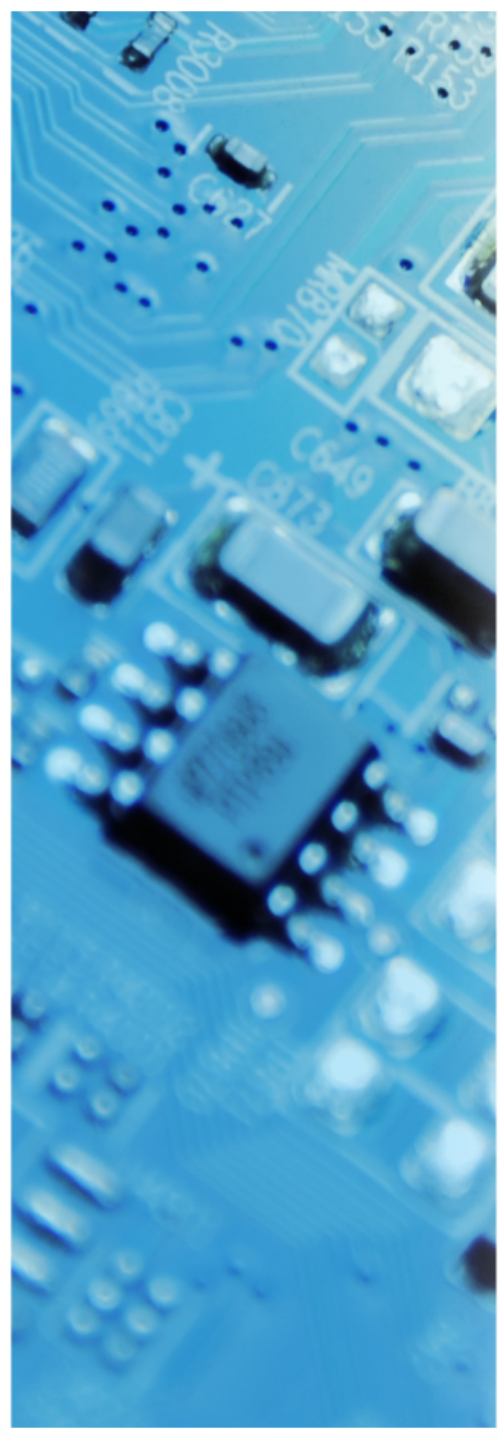


Métodos de busca



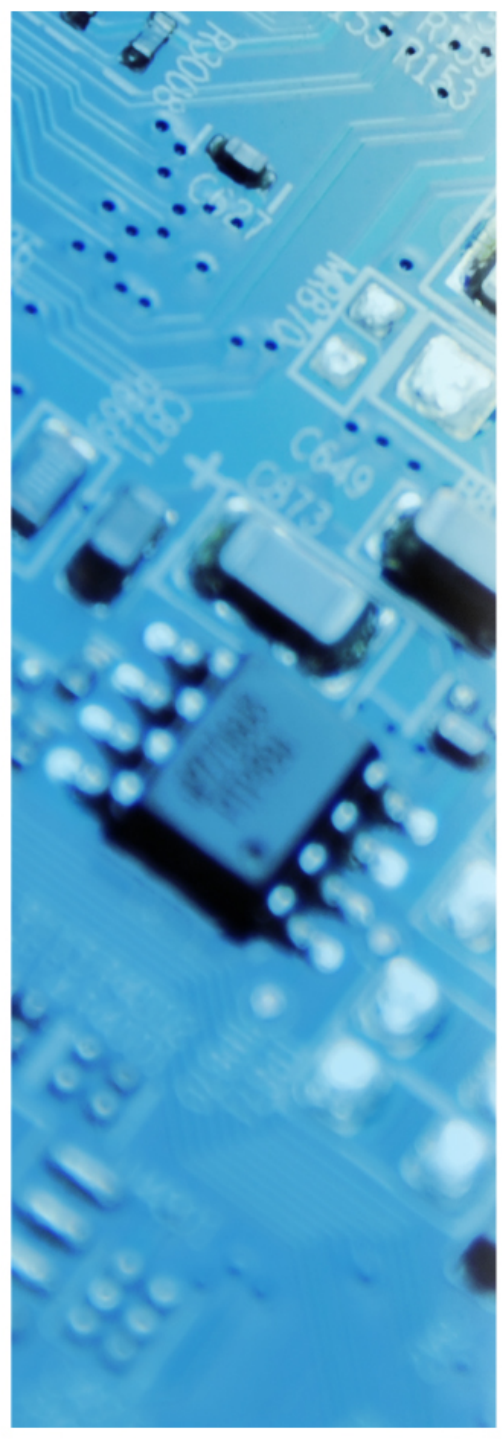
Métodos de busca

- Método de busca é um termo comum na área de computação.
- Pode ser usado para definir algoritmos que buscam respostas para problemas, como busca em largura, profundidade, algoritmos genéticos, entre outros.
- O termo também é utilizado para definir algoritmos mais simples, que pesquisam a existência de um elemento dentro de um vetor, lista, etc.



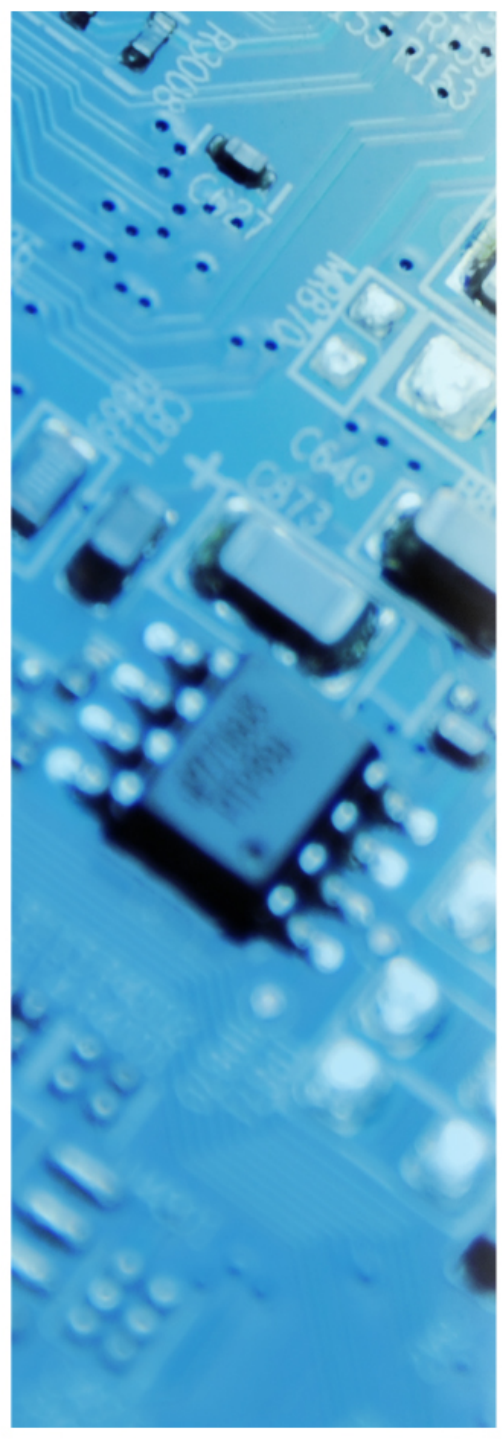
Métodos de busca

- Dentro deste contexto, vamos considerar a seguinte definição:
- Um algoritmo de busca recebe uma lista de valores e um valor a ser encontrado e retorna a(s) posição(s) na qual o valor está ou um indicativo que o valor não se encontra na lista.
- A definição quanto ao retorno somente de uma posição (primeira) ou das diversas posições em que o valor for encontrado depende da implementação do método e da necessidade da aplicação.



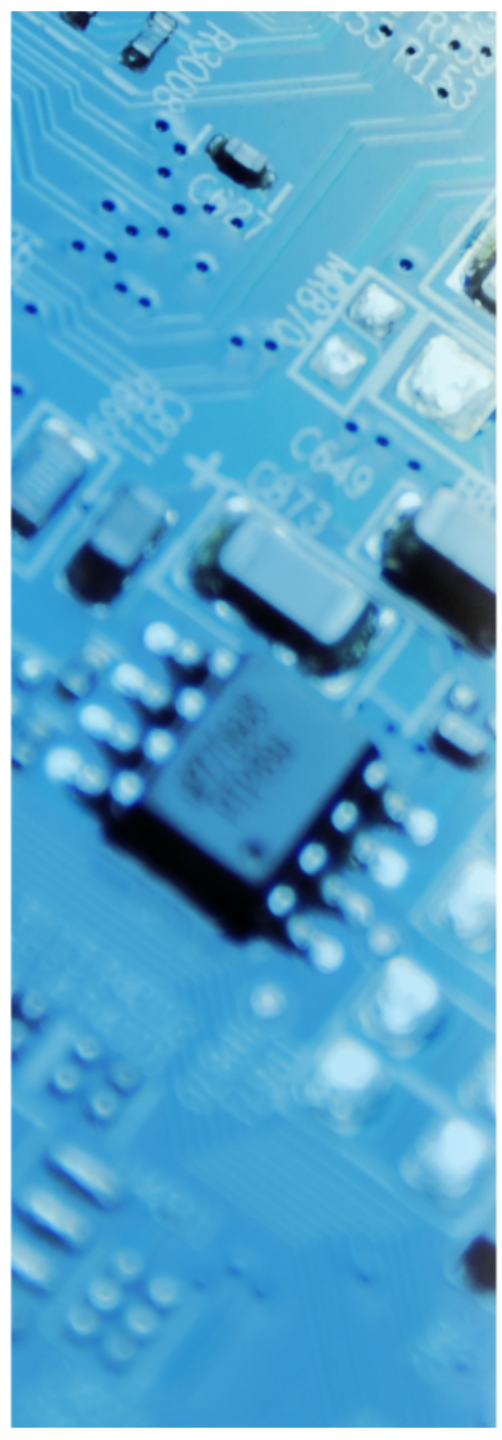
Métodos de busca

- Assim como a ordenação, o problema de busca é um dos problemas básicos da área de computação.
- A grande maioria das linguagens possui implementações como métodos ou funções aplicáveis a determinados objetos.
- Entretanto, em várias situações é necessário implementar uma versão adaptada ao problema.
- Em muitos casos a indicação se serão buscadas as diversas ocorrências é um parâmetro da função ou método.



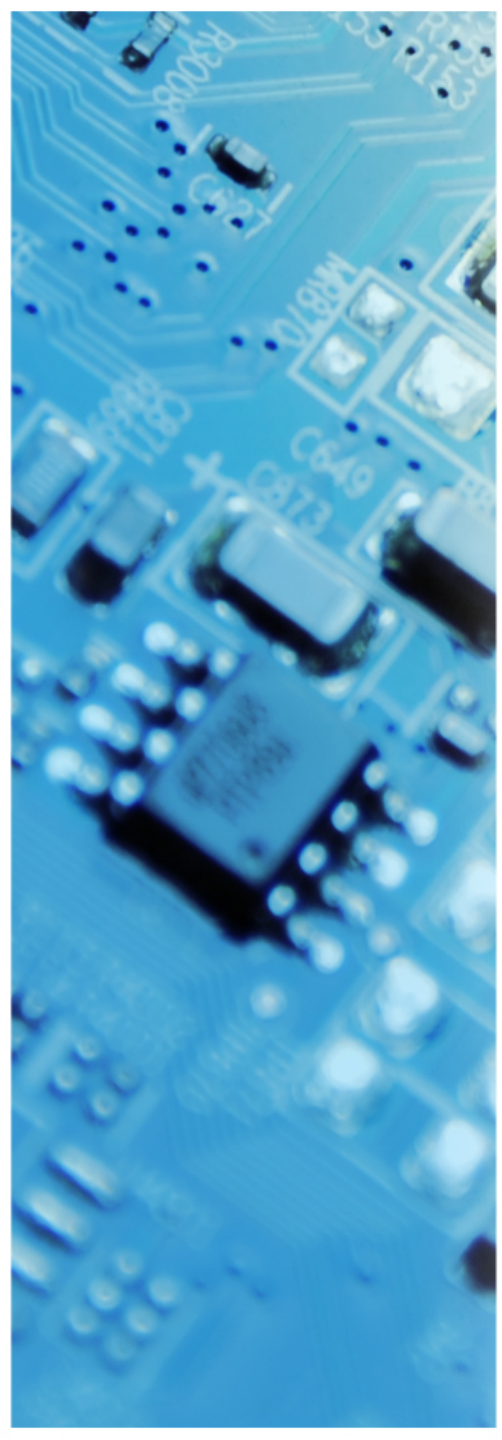
Métodos de busca

- Na maioria dos casos, métodos de busca são implementados para pesquisar valores numéricos, como inteiros, por exemplo.
- Entretanto, é possível aplicar a busca a qualquer tipo de dado.
- Se houver uma chave, um valor que pode ser comparado em uma operação de igualdade os métodos podem ser empregados.



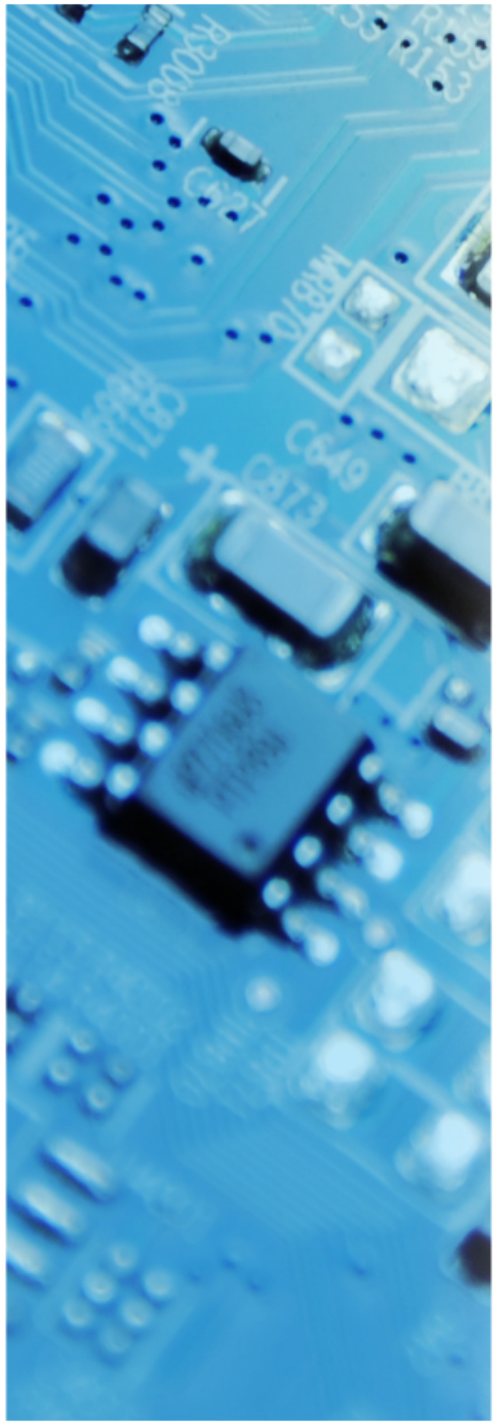
Métodos de busca

- Exemplos de aplicação:
  - Procurar uma pessoa pelo seu código.
  - Procurar um produto pelo seu nome.
  - Buscar quais vendas foram realizadas com valores iguais a 100.
  - Buscar itens de venda com preço igual a 2.50



Métodos de busca

- Problema: Criar uma função denominada busca(lista, chave)
  - Parâmetro lista é um objeto do tipo array, list, set ou outro.
  - Parâmetro chave é um valor do mesmo tipo dos elementos armazenados na lista.
  - O retorno da função é a(s) posição(s) da chave na lista ou -1, caso a chave não tenha sido encontrada.



Métodos de busca

- Exemplo 1: busca(lista, 50)

5	15	35	40	50	78	80	90	102	104

res=4

- Exemplo 2: busca(lista, 100)

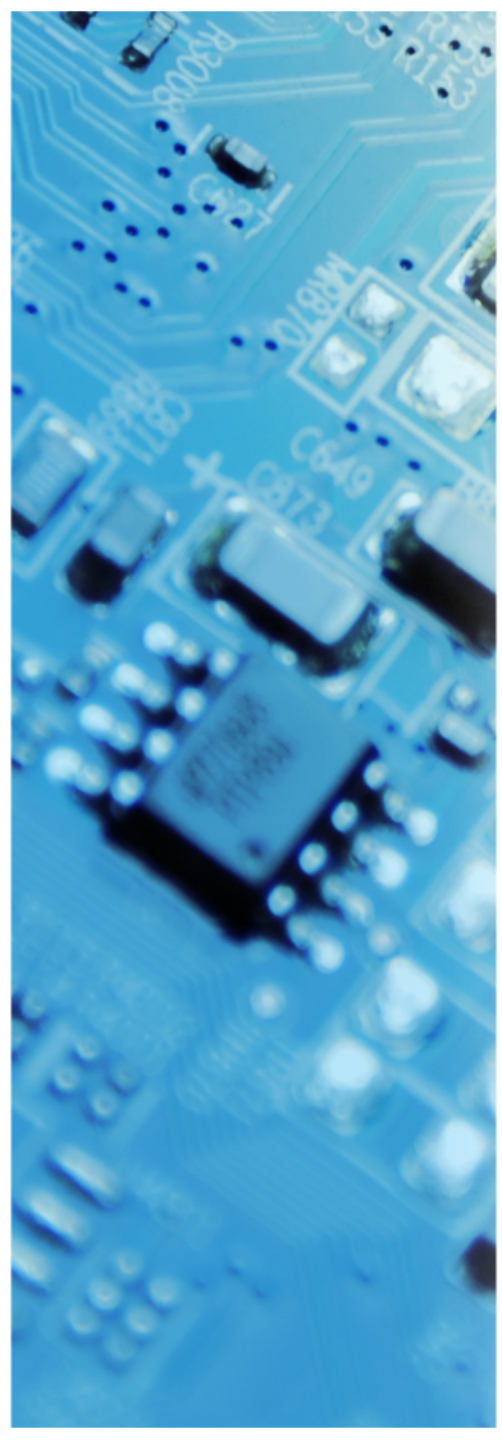
100	90	87	34	12	26	100	80	87	26

res=[0,6]

- Exemplo 3: busca(lista,33)

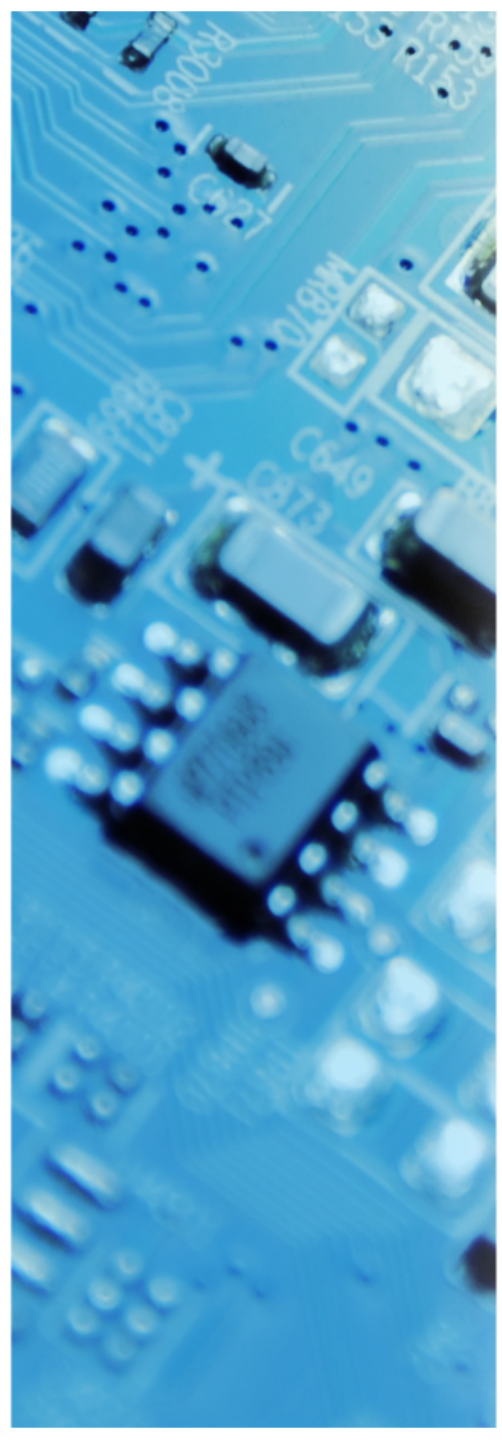
8	11	23	45	56	78	90	101	102	110

res=-1



Métodos de busca

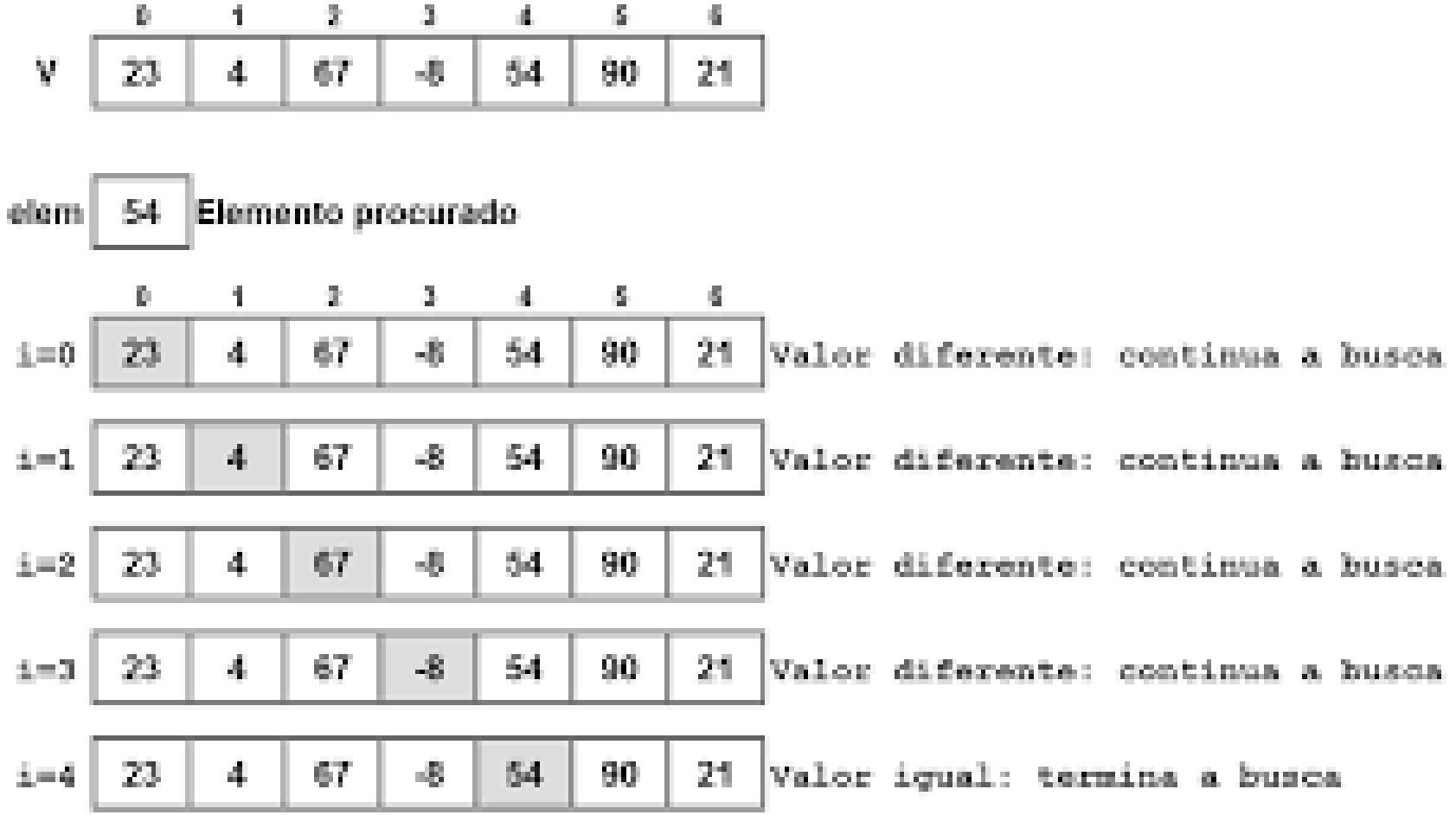
- É possível encontrar diferentes estratégias para realizar a busca.
- As duas mais conhecidas são a busca linear e a busca binária.
- Buscas que utilizam outras condições, como >,<,>=,<= ou <> podem ser consideradas variações do mesmo problema.
- Em SGBDs é possível encontrar diversos algoritmos aprimorados para a busca.
- Estes algoritmos são utilizados na execução de consultas SQL.

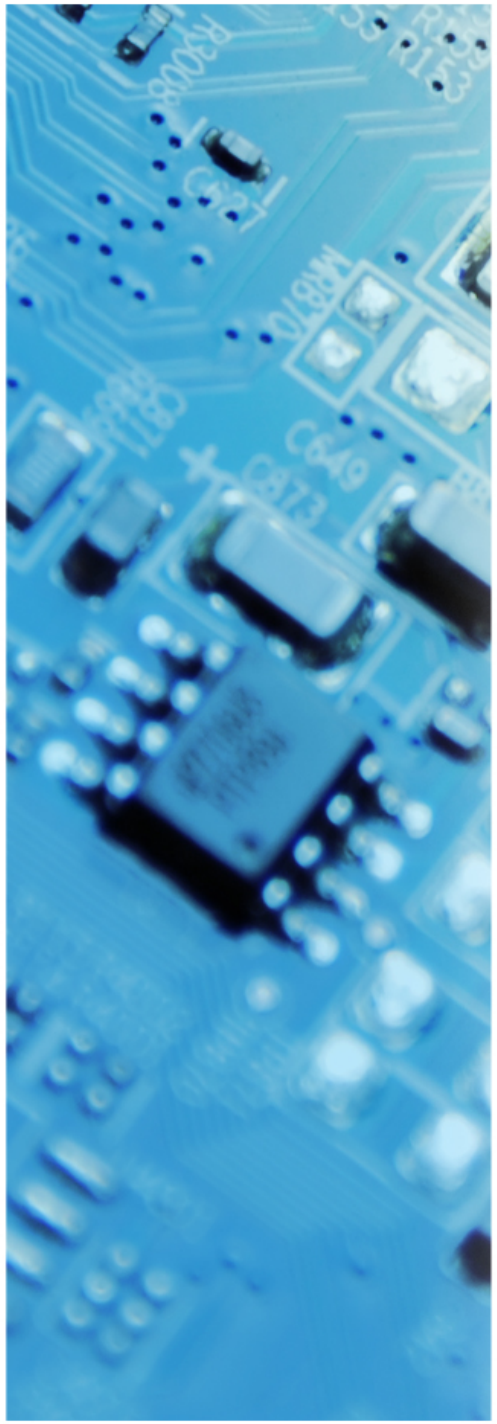


Busca sequencial

- O algoritmo de busca sequencial, também chamada de busca linear é o método mais simples.
- A lista é percorrida e cada valor armazenado é comparado com a chave.
- Se o valor é encontrado, a posição é retornada.
- A busca se encerra caso o objetivo seja somente saber se a chave existe, caso contrário, a busca continua e as demais posições são retornadas.
- Ao final, caso nenhuma comparação tenha sido verdadeira, o retorno é -1.
- Este método não requer uma lista de valores ordenados, uma vez que todos são lidos e comparados.

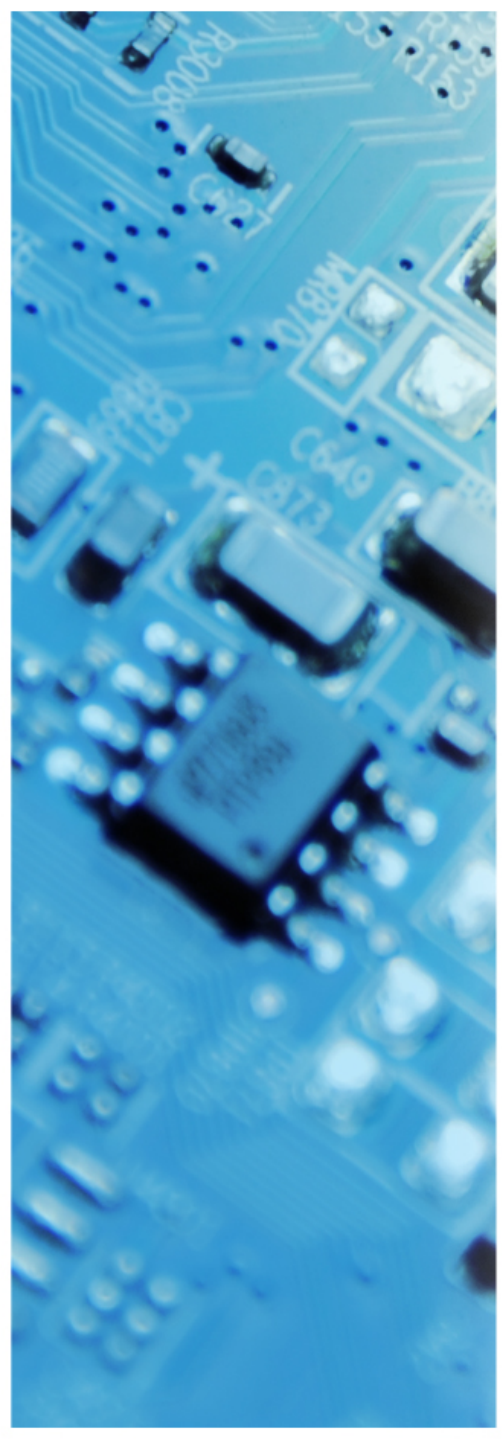
Busca sequencial





Busca binária

- O método de busca binária é mais sofisticado e eficiente que a busca sequencial, porém, requer uma lista ordenada para o seu correto funcionamento.
- Este método se baseia na divisão da lista em partes menores.
- Os princípios se assemelham à arvores balanceadas, nas quais os valores menores ficam a esquerda e os maiores a direita.
- Como a lista se encontra ordenada é possível partir do valor central, buscando a esquerda ou direita conforme o caso.

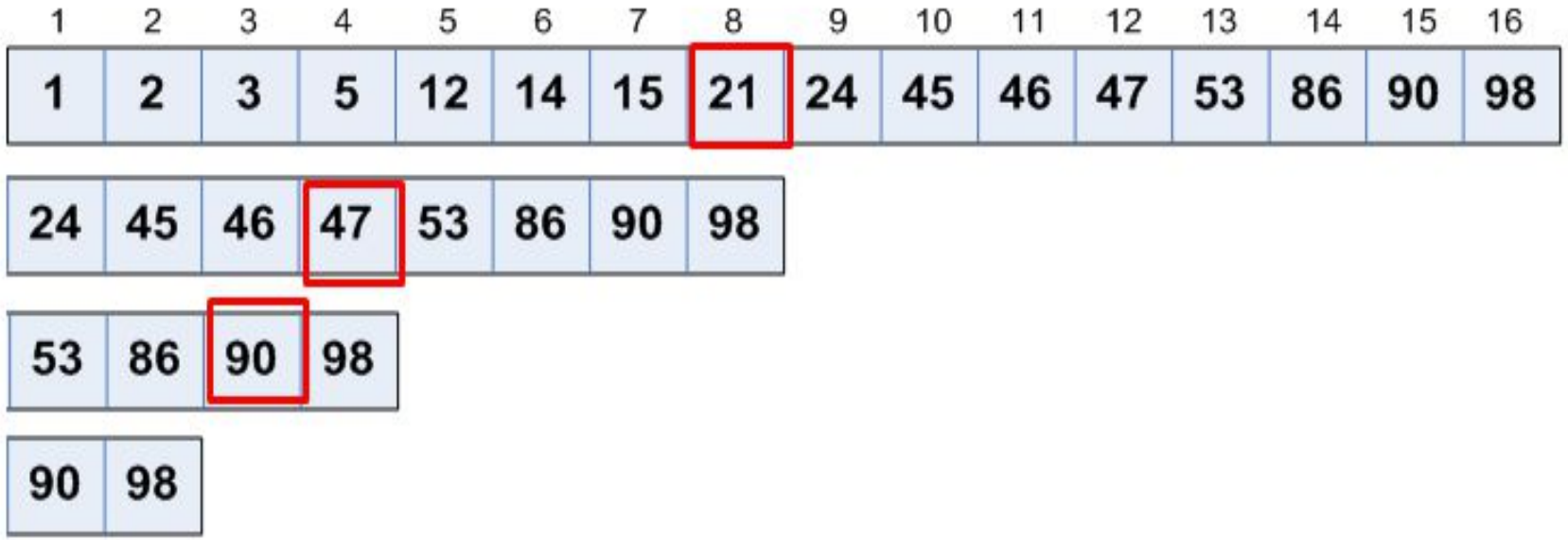


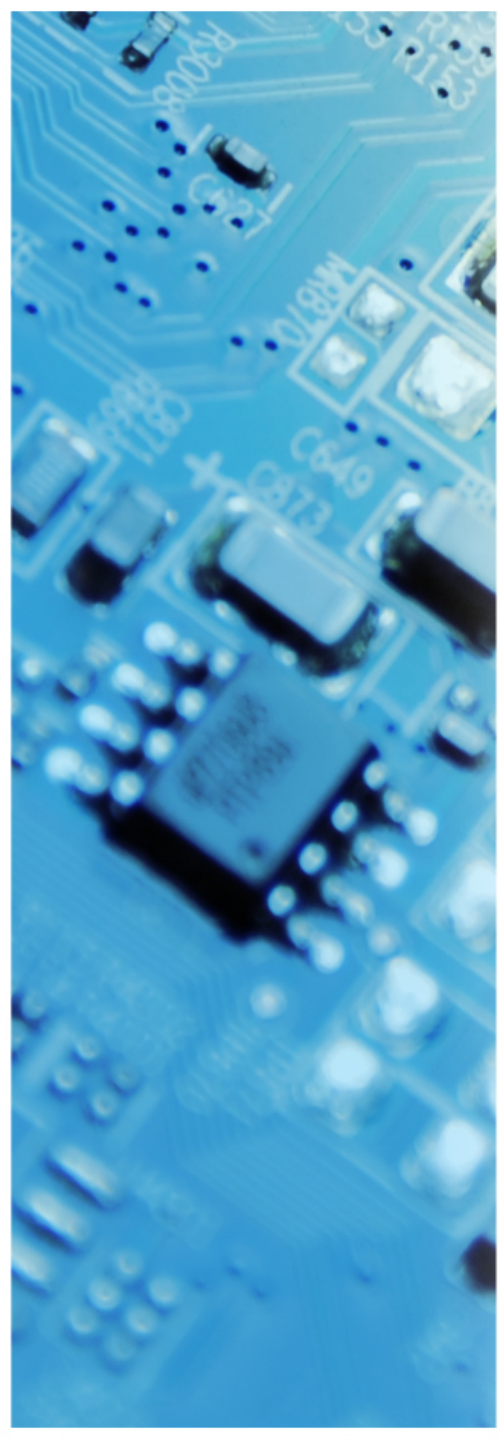
Busca binária

- Passos da busca:
  - Compara o valor buscado com o elemento centra da lista.
  - Caso o valor seja igual ao armazenado nesta posição, retorne a posição e encerre a busca.
  - Caso contrario, se o valor for menor repita o processo na primeira metade, da posição 0 até a atual.
  - Caso seja maior, repita o processo na segunda metade, da posição atual até a posição final.
  - Encerra a busca caso não seja encontrado ou caso não seja mais possível dividir a lista pesquisada.

Busca binária

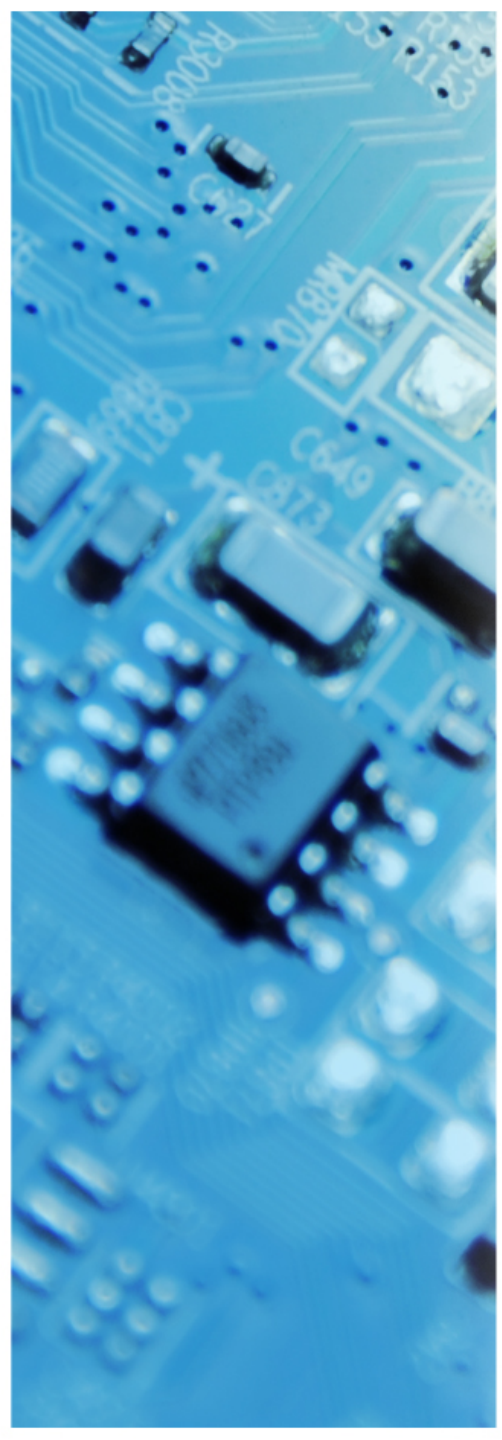
- Valor buscado: 90





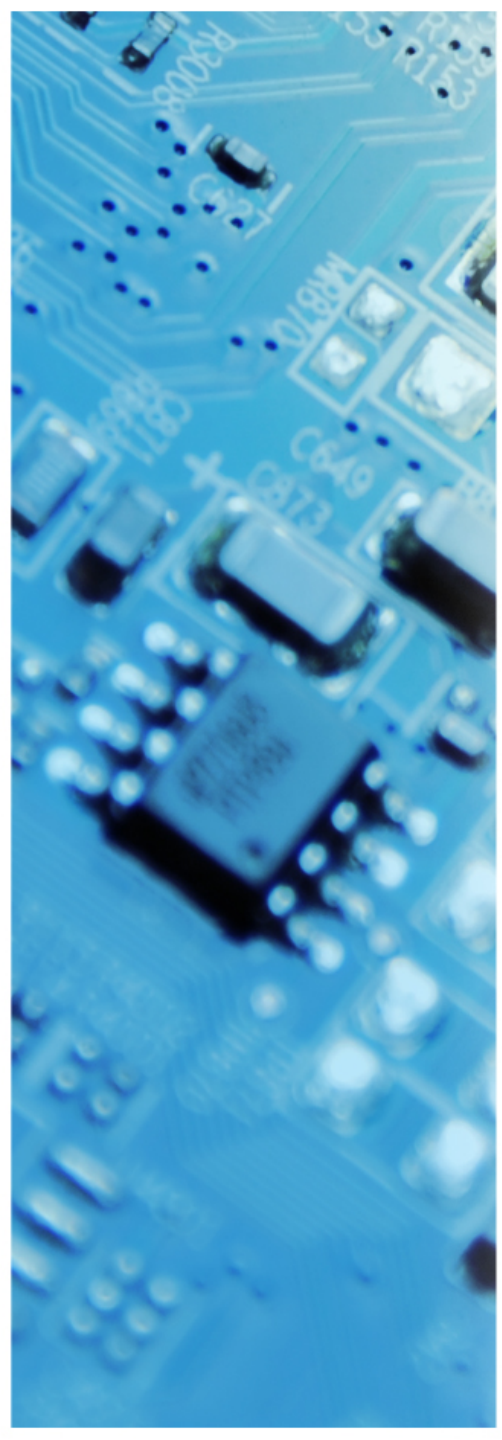
Eficiência dos métodos

- No algoritmo linear, na melhor hipótese o valor buscado está na posição 0.
- Na pior hipótese ele se encontra no final da lista.
- Neste caso o tamanho da lista é decisivo para uma maior eficiência do método.
- Se todas as chaves possuem a mesma probabilidade de serem buscadas o cálculo da eficiência para diversas buscas seria dado pela fórmula abaixo.
- (n+1)/2, sendo n o número de elementos da lista



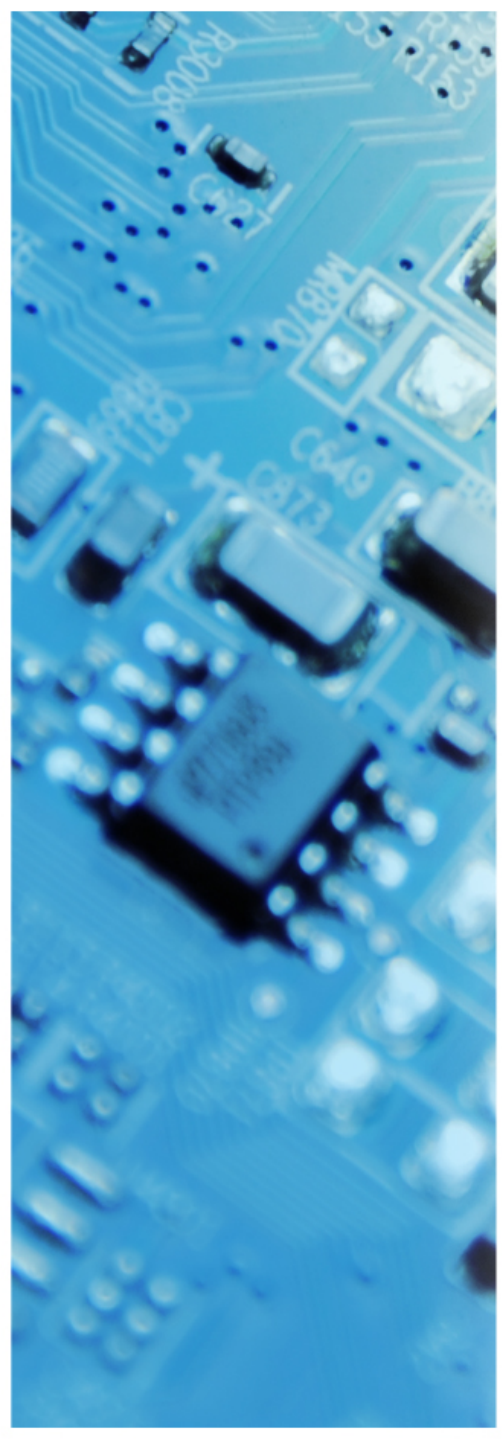
Eficiência dos métodos

- Na busca binária, na melhor hipótese a chave se encontra na posição central da lista.
- Na pior hipótese será necessário dividir a lista até que o tamanho fique igual a 1.
- A cada operação a lista fica menor.
- Desta forma o cálculo se baseia na seguinte questão: Quantas vezes é possível dividir a lista?
- Esta definição nos leva a fórmula do logaritmo de base 2, ou seja, na pior das hipóteses o número de acessos é dado por $\log_2(n)-1$.



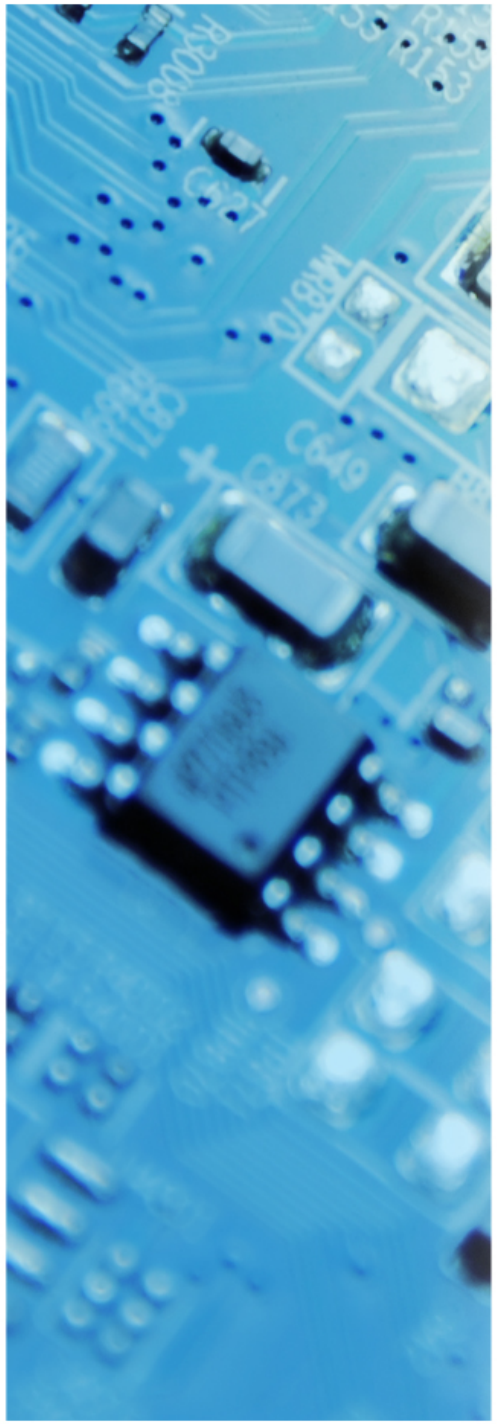
Eficiência dos métodos

- Uma comparação possível seria a busca de uma chave em uma lista que possui um milhão de valores (10^6).
- Busca sequencial ou linear: 10^6, ou seja, 500000 acessos
- Busca binária: $\log_2(10^6)$, ou seja 19 acessos.
- Em ambos os casos considera-se valores em média, ou seja, para valores que possuem a mesma probabilidade de serem buscados



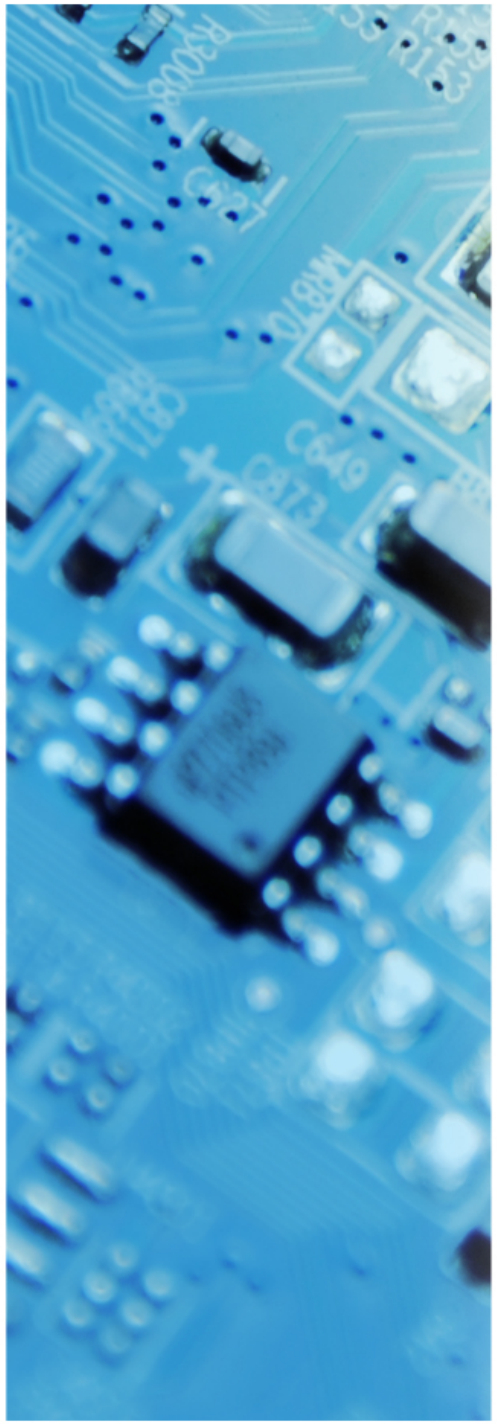
Eficiência dos métodos

- Uma questão importante é que na busca binária os dados devem estar ordenados.
- Se uma lista sofre muitas mudanças, retiradas, inclusões, pode ser necessário ordenar ela com frequência.
- Se houver a necessidade de ordenar sempre que a busca for feita, a eficiência do método pode ser comprometida.
- Técnicas para inserção e retirada mantendo os dados ordenados podem ser usadas.



Desafio 1

- Pesquisar implementações de métodos de busca linear e binária em python (pergunte ao ChatGPT).
- Testar os métodos para uma lista 10000 valores inteiros.
- Gerar listas com range de 0 a 10000, por exemplo.
- Neste caso as listas estarão ordenadas, pois os valores serão gerados do menor para o maior.
- Executar várias buscas (no mínimo 5), sendo que em cada busca será gerado um valor randômico para ser encontrado.
- Executar pelo menos duas buscas com valores que não existem, que estão fora do limite.
- Verificar os tempos de execução e a quantidade de testes feitos em cada método, em cada uma das buscas.



Desafio 2

- Alterar a solução desenvolvida na aula anterior, adicionar uma opção para buscar e mostrar se um determinado PET foi atendido.
- Implementar a opção com o método de busca binária.
- Neste caso, os atendimentos estarão em uma lista que deverá estar ordenada pelo código do pet.
- O usuário irá informar o código do PET e o sistema irá buscar se existe atendimento e retornar a posição do atendimento na lista, caso não encontre irá retornar -1.