

Buscas em Array

Técnicas de Programação - 2025/02

Última atualização: 19/08/2025 10:45

Parte 1 - buscas simples

Vamos começar com algumas buscas bem simples. Esse é mais um exercício de escrever pseudo-código do que de pensar de fato.

Problema

Devolver o índice valor 1 mais à esquerda em vetores com o abaixo.

	0	1	2	3	4	5	6
Array A	0	0	0	1	1	1	1

Em geral, tendemos a iterar em arrays começando pelo índice 0. Faça a busca começando pelo início.

Algorithm 1: BuscaInicio

Input : array A

Output: int

Uma outra possibilidade é começar pelo fim do array. Escreva este busca no espaço abaixo.

Algorithm 2: BuscaFim

Input : array A **Output:** int

Parte 2 - busca binária

Nesta parte iremos formalizar o algoritmo da busca binária. A ideia básica foi:

Busca binária

1. seleciona o elemento na posição exatamente no meio do vetor
2. se esse elemento for 0, continua procurando na metade da direita do vetor
3. se esse elemento for 1, continua procurando na metade da esquerda do vetor
4. repete esse procedimento até encontrar um elemento que é 1 e tem um vizinho que é 0.

Nosso processo de concretização desses passos será por meio de simulações. Iremos fazer um passo a passo e preencher o que o algoritmo **deveria** fazer. Então iremos usar essas simulações para escrever um algoritmo que realize esse passo a passo.

Revisão

Marque abaixo os arrays em que poderíamos usar a busca binária.

- [0, 0, 0, 1, 0, 1]
- [1, 0, 0, 1, 1, 1]
- [0, 0, 0, 0, 0, 0]
- [0, 0, 1, 1, 1, 1]
- [0, 0, 0, 4, 6, 7, 10]
- [2, 1, 4, 7, 4, 8, 10]

Agora partiremos para simulação no array $A = [0, 0, 0, 0, 0, 1, 1]$. Vamos iniciar com uma ideia simples: Desenhe o array abaixo e execute os passos 1, 2 e 3 da ideia da Busca binária. Coloque então um X em todas as posições **eliminadas** do array e escreva o índice do elemento do meio usado no passo 1.

Com isso, temos a ideia de procurar em um **intervalo** de A. No caso acima, concluímos que a busca será no intervalo $[4, 7)$.

Intervalos

Todo intervalo é aberto no fim. Ou seja, o intervalo $[1, 5)$ contém os elementos 1, 2, 3, 4 mas não contém o 5.

Agora volte no seu desenho acima e faça o processo novamente, mas considerando o início do array como o índice 4 e o fim como o índice 7. Qual seria o novo meio? E quais elementos seriam descartados?

Calculando o meio e os novos intervalos

Conseguir “adivinhar” a expressão para calcular o meio não é tão simples assim. Vamos então construir uma tabela com valores de início e fim e calcular o meio manualmente. Isso, em geral, ajuda a encontrar padrões e deve nos guiar a descobrir a expressão correta. As duas primeiras linhas já estão preenchidas. Complete o restante da tabela.

início	fim	N elementos	meio
0	5	5	2
0	6	6	2
1	4		
1	5		
7	12		
5	6		
0	1		

Com base nos valores acima, escreva abaixo a expressão usada para calcular o número de elementos e o meio. Quando terminar, valide com o professor, alguns dos ninjas ou um colega que já tenha validado a solução.

Determinar quais é a parte do array que vamos manter na busca é o próximo passo. Complete a tabela abaixo. Como antes, já temos algumas linhas preenchidas.

início	fim	A[meio]	novo início	novo fim
0	5	0	3	5
0	6	1	0	3
1	4	1		
1	5	0		
7	12	1		
5	6	0		
0	1	1		

Quando $A[\text{meio}] = 0$, quais as expressões para calcular o novo intervalo? Escreva em termos de `inicio`, `fim` e `meio`.

E quando $A[\text{meio} = 1]$?

Algoritmo completo

Com todas essas ideias juntas, simule o algoritmo completo para o array

$A = [0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1]$.

O algoritmo sempre começa com o array todo (`inicio=0` e `fim=A.length`).

inicio	fim	N	meio	A[meio]
0	9	9		

Quando o algoritmo para?

Na tabela acima, quando paramos de dividir o array? Escreva a expressão dentro do box.

Agora vamos, finalmente, tentar escrever o pseudo código. No fim da aula ele será escrito na lousa. O importante aqui é tentar, o máximo possível, transformar os passos da simulação em passos válidos como pseudo código.

Algorithm 3: BuscaBinariaArray01

Input : array A **Output:** int

Reflexão

Nosso algoritmo funciona para arrays com somente 0 e 1. Como poderia generalizar para buscar pelo valor 4 no array $A=[1\ 4\ 7\ 10\ 44\ 100]$?