Estrutura de Dados I Exercício

1 Busca Binária

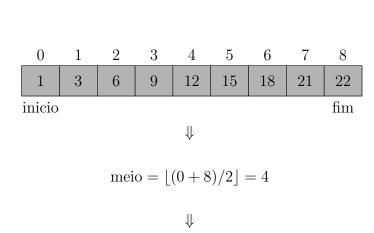
O problema da busca em um vetor ordenado pode ser descrito da seguinte forma:

<u>Problema da busca em um vetor ordenado:</u> dados um vetor de números inteiros V ordenado de forma não decrescente e um inteiro x, encontre a posição k de V, tal que $V[k] < x \le V[k+1]$.

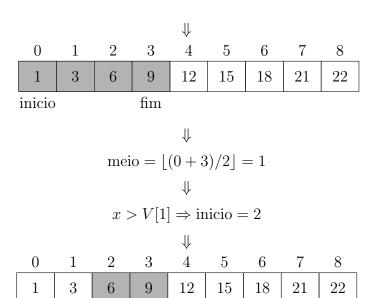
A busca binária (ou também chamada de pesquisa binária) realiza uma busca por um elemento em um vetor ordenado. O algoritmo funciona dividindo repetidamente pela metade a porção da lista que deve conter o item, até reduzir as localizações possíveis a apenas uma.

Exemplo: Seja V o vetor com nove números inteiros ilustrado a seguir.

- Queremos encontrar a posição de x = 9.
- Precisamos encontrar x no vetor V, com posições no intervalo de 0 a 8.
- As variáveis inicio e fim guardam a primeira e última posição do segmento do vetor que será analisado.

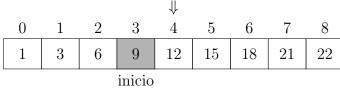


$$x < V[4] \Rightarrow \text{fim} = 3$$



inicio fim

$$x > V[2] \Rightarrow \text{inicio} = 3$$



fim

$$\text{meio} = \lfloor (3+3)/2 \rfloor = 3$$

 $x = V[3]! \Rightarrow$ A posição 3 contém o valor x!

Vamos comparar o valor de x com o elemento que está no meio do vetor. Para isso, vamos calcular a posição do meio.

O meio do vetor está na posição 4, então comparamos o valor de x com o valor que está na posição 4. Temos que x < V[4], então podemos descartar a busca de x nas posições de 5 a 8 de V. Assim, realizamos a busca por x na primeira metade de V.

Ou seja, vamos procurar x no segmento de V com inicio = 0 e fim = 3.

Calculamos a próxima posição que iremos analisar. No caso, meio = 1.

Como x > V[1], vamos procurar x no segmento com inicio = 2 e fim = 3.

Vamos calcular a posição do "meio" do segmento de V que será analisado.

Como x > V[2], vamos procurar x na metade do segmento atual com inicio = 3 e fim = 3.

Calculamos o valor do meio do segmento e temos que meio = 3.

Comparando o valor na posição 3 de V, temos que x=V[3]. Logo, encontramos x na posição 3 de V!

Questão 1: Algoritmo Iterativo de busca binária

- a) Implemente o algoritmo iterativo de busca binária em um vetor de números inteiros, ordenado de forma não decrescente, com tamanho N.
- b) Explique em seu código (em formato de comentário) como você garante que seu algoritmo termina.
- c) Implemente a função main.c para testar a função de busca implementada. A função main deve alocar dinamicamente o vetor de inteiros com o tamanho fornecido pelo usuário e preenchê-lo com valores inteiros informados pelo usuário. Após o preenchimento do vetor, a função deve imprimir o vetor e solicitar ao usuário que digite um valor para a busca, realizar a busca binária e imprimir a posição do elemento no vetor, caso ele exista, e a mensagem: "O elemento informado não existe no vetor."
- d) Calcule a complexidade total de cada linha da função de busca binária em uma execução da função. Ao final apresente e explique os cálculos da complexidade da função e a ordem assintótica da função resultante. As complexidades podem ser escritas no próprio arquivo de código como comentários do código.

Questão 2: Algoritmo Recursivo de busca binária

- a) Implemente o algoritmo recursivo de busca binária para um vetor de números inteiros, ordenado de forma não decrescente, com tamanho N.
- b) Indique no seu código (em formato de comentário) qual é o caso base do algoritmo e mostre que o algoritmo sempre converge para o caso base.
- c) Indique e explique no seu código (em formato de comentário) qual é o passo recursivo. Explique qual é o subproblema da chamada recursiva.
- d) Implemente a função main.c para testar a função de busca implementada. A função main deve alocar dinamicamente o vetor de inteiros com o tamanho fornecido pelo usuário e preenchê-lo com valores inteiros informados pelo usuário. Após o preenchimento do vetor, a função deve imprimir o vetor e solicitar ao usuário que digite um valor para a busca, realizar a busca binária e imprimir a posição do elemento no vetor, caso ele exista, e a mensagem: "O elemento informado não existe no vetor."