Estrutura de Dados

Busca Seqüencial e Binária

Profa.: Márcia Sampaio Lima

EST - UEA

Vetores

- Busca ou Pesquisa:
  - Dado um inteiro x e um vetor **V**[1..n] de inteiros, o problema da busca consiste em encontrar x em **V**.
    - Encontrar um índice k tal que **V**[k] == x.
    - O problema faz sentido com qualquer n > 0.
    - Se n é 0, o vetor é vazio e portanto essa instância do problema não tem solução.

Vetores

- Pesquisa seqüencial (PS) ou linear
  - Uma solução possível é percorrer o vetor desde a primeira posição até a ultima.
  - Para cada posição, comparamos vetor[i] com valor.
    - Se forem iguais dizemos que valor existe.
    - Se chegarmos ao fim do vetor sem sucesso dizemos que valor não existe.

Vetores

- Pesquisa Seqüencial (PS)
  - **Metodologia**: Percorre o vetor, **elemento por elemento**, verificando se o elemento desejado está presente no vetor.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
14	**21**	**5**	**45**	**12**	**3**	**86**	**98**	**46**	**53**	**24**	**2**	**1**	**15**	**90**	**47**

Pergunta: Como verificar se o **elemento 90** está presente no vetor acima?

Pergunta: Quantas comparações são necessárias para achar o **elemento 90**?

Vetores

- Pesquisa Seqüencial:
  - Características:
    - **Extremamente simples** o algoritmo;
    - Pode ser **muito ineficiente** quando o conjunto de dados é **muito grande**.

Vetores

- Busca Seqüencial: desempenho computacional
  - **Pior Caso**: Qual o cenário de pior caso possível ?

    É quando é necessário realizar n comparações (onde n é o número de elementos);

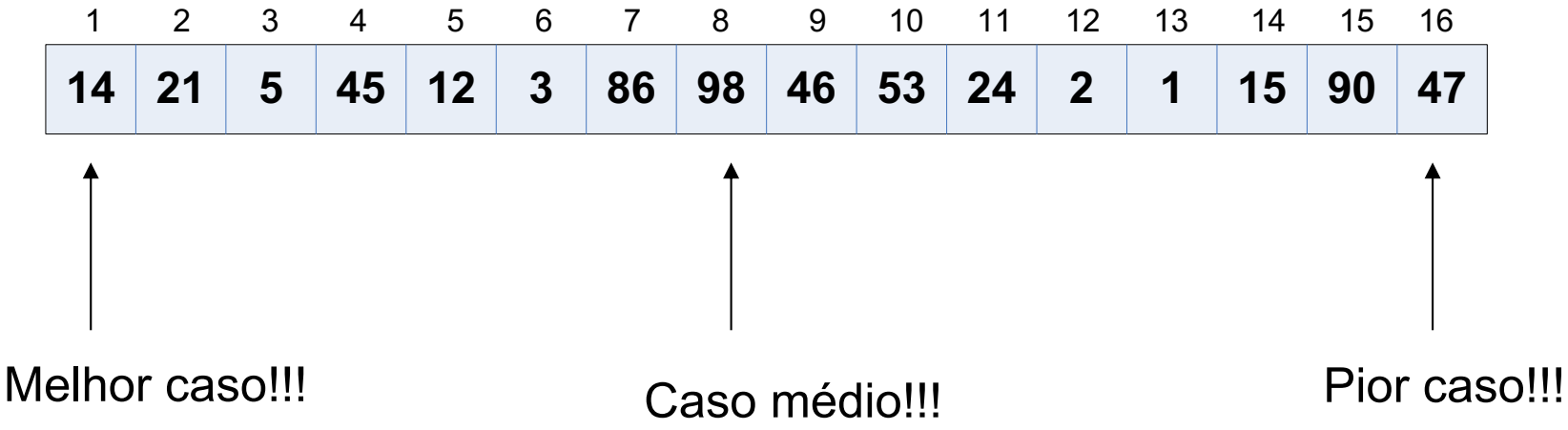
  - **Melhor Caso**: Qual o cenário de melhor caso possível ?

    É quando é necessário realizar somente uma comparação.

  - **Caso Médio**: Qual o cenário de caso médio possível ?

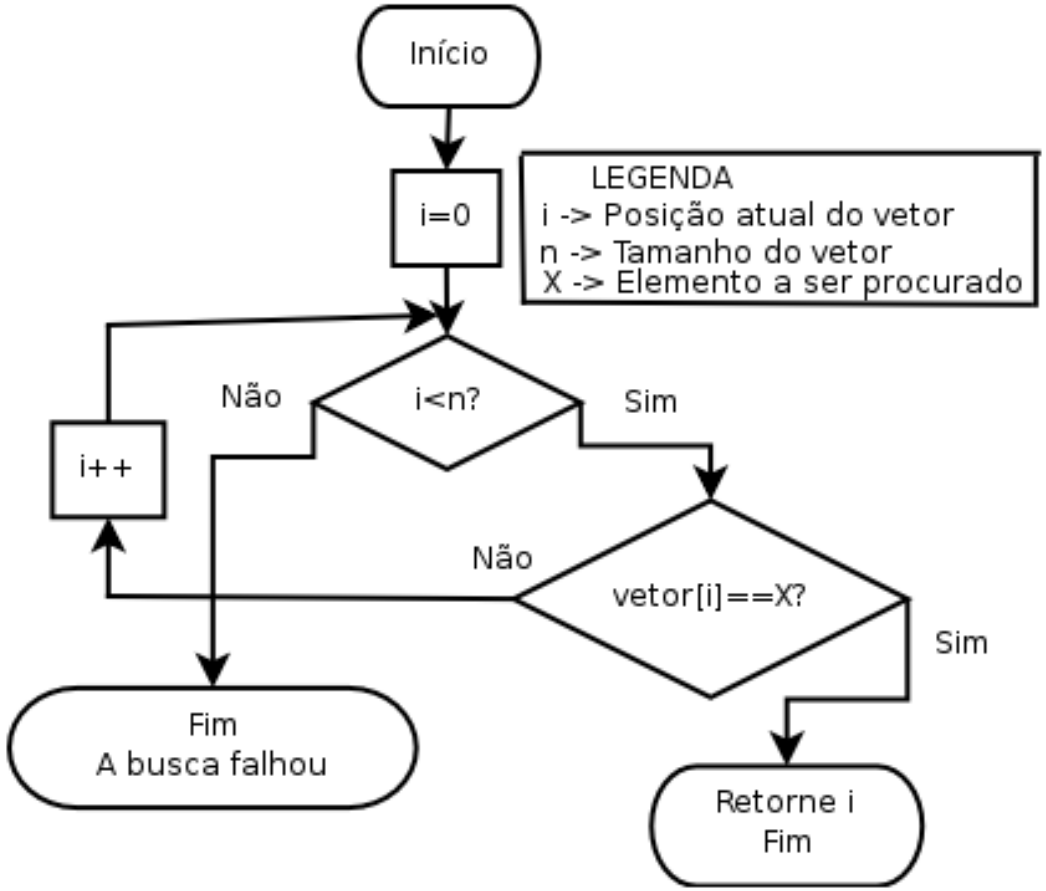
    - É quando é necessário realizar **cerca** de n/2 comparações.

Vetores


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
14 21 5 45 12 3 86 98 46 53 24 2 1 15 90 47
Melhor caso!!!
Caso médio!!!
Pior caso!!!

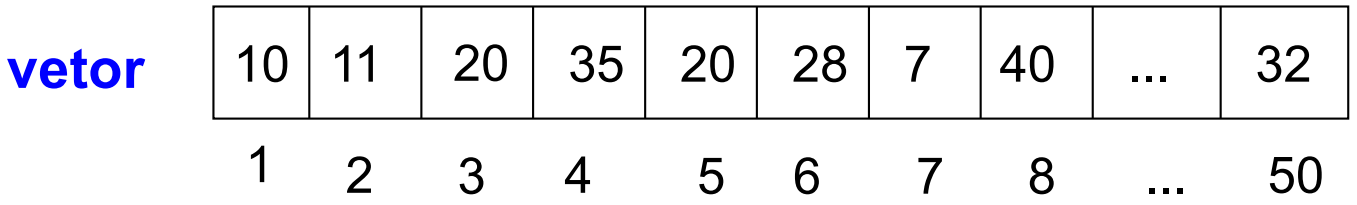
Vetores

- Fluxograma Busca Seqüencial:



Vetores

- Exercício:
  - Faça um programa em C que leia e armazene um conjunto de 50 temperaturas inteiras. Escreva em quais dias ocorreu uma determinada temperatura a ser informada pelo usuário.



Temperatura 20

saída 3, 5

```
#include <iostream>
using namespace std;
int main(){
   int temperatura[] =
                  {30,8,38,20,24,91,22,33,44,78,20};

   for(int i = 0; i < sizeof(temperatura); i++){
      if (temperatura[i] == 20){
          cout << (i+1);   }
   }
return 0;
}
```

```
#include <iostream>
using namespace std;
int main(){
   int temperatura[] =
                  {30,8,38,20,24,91,22,33,44,78,20};

   for(int i = 0; i < sizeof(temperatura); i++){
      if (temperatura[i] == 20){
          cout << (i+1);   }
   }
return 0;
}
```

O(n)

Vetores

- Busca Seqüencial:
  - Seria possível melhorarmos a eficiência do método apresentado?
  - Como !?

Vetores

- Pesquisa Binária (PB):
  - Forma mais **eficiente** de realizar pesquisas em relação ao método de PS.
  - **Metodologia**:
    - Consiste em comparar **alguns itens** do vetor com o dado (**chave alvo**) que deseja-se encontrar.
    - **Premissa**: os dados contidos no vetor já estão ordenados segundo um critério.

Vetores

- Pesquisa Binária (PB):
  - Passos do processo:

    1) Checar onde está o **ponto médio** do vetor.

    2) Comparar o **elemento do ponto médio** (EPM) com a **chave alvo** (CA).

    3) Caso não encontre o dado no passo 2, continuar a pesquisa da seguinte forma:
      - Caso CA**<**EPM realizar a pesquisa no sub-vetor a esquerda do EPM, partindo do **passo 1**.
      - Caso CA**>**PM realizar a pesquisa no sub-vetor a direita do EPM, partindo do **passo 1**.
      - Caso CA=EPM, então a pesquisa para com sucesso, pois achou o dado desejado!

Vetores

- Pesquisa Binária (PB):
  - Passos do processo:

    1) Checar onde está o **ponto médio** do vetor.

    2) Comparar o **elemento do ponto médio** (EPM) com a **chave alvo** (CA).

    3) Caso não encontre o dado no passo 2, continuar a pesquisa da seguinte forma:
      - Caso CA**<**EPM realizar a pesquisa no sub-vetor a esquerda do EPM, partindo do **passo 1**.
      - Caso CA**>E**PM realizar a pesquisa no sub-vetor a direita do EPM, partindo do **passo 1**.
      - Caso CA=EPM, então a pesquisa para com sucesso, pois achou o dado desejado!

Vetores

- Busca Binária:

Exemplo Inicial:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
14	**21**	**5**	**45**	**12**	**3**	**86**	**98**	**46**	**53**	**24**	**2**	**1**	**15**	**90**	**47**

Após ordenação: !!!???

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	**2**	**3**	**5**	**12**	**14**	**15**	**21**	**24**	**45**	**46**	**47**	**53**	**86**	**90**	**98**

Pergunta: Como verificar se o **elemento 90** está presente no vetor acima?

Pergunta: Quantas comparações são necessárias para achar o **elemento 90**?

Pergunta: Como verificar se o **elemento 71** está presente no vetor acima?

Pergunta: Quantas comparações são necessárias para achar o **elemento 71**?

Vetores

- Qual dos dois algoritmos é o melhor?
  - O algoritmo de pesquisa binária assume que o vetor está ordenado. Ordenar um vetor também tem um custo, superior ao custo da pesquisa seqüencial.
  - Se for para fazer **uma** só pesquisa, não vale a pena ordenar o vetor. Por outro lado, se pretendermos fazer muitos pesquisas, o esforço da ordenaçãoo já poderá valer a pena.

Vetores

- Exercício:
  - Implementar em C um programa que efetue busca binária em um vetor de 900 números inteiros.
    - Preencha o vetor com números inteiros aleatórios.

```
#include<stdio.h>
#include <iostream>
using namespace std;
int main() {
    int temperatura[] = {8,20,22,24,91,122,133,144,178,220};
    int inicio=0, fim, meio, busca = 122;

    fim = sizeof(temperatura)/sizeof(int);

    while(inicio <= fim){
       meio = (inicio + fim)/2;
       if (temperatura[meio] == busca){
          cout << meio << endl;
          break;   }

      if (busca > temperatura[meio]) {
         inicio = meio+1;   }
      Else { fim = meio -1;   }

   }
   return 0;
}
```

O(log n)