Aula 8: Busca Binária

- Princípio
- Algoritmo de Busca Binária
- Complexidade

Princípio

- A Busca Binária é empregada apenas em Listas Ordenadas.
- Princípio: Percorrer a tabela como se folheia uma lista telefônica, isto é:
  - Abrir a lista no meio;
  - Se a informação procurada encontra-se antes da página que estamos olhando, passamos a procurá-la na primeira metade da lista, desprezando a segunda metade;

Princípio

- Se, pelo contrário, a informação encontra-se depois da página que estamos olhando, passamos a procurá-la na segunda metade da lista, desprezando a primeira metade;
- Abrimos a lista novamente, no meio da metade escolhida, e assim sucessivamente.

Princípio

Exemplo: Vamos procurar a chave $x = 19$ na lista com 16 elementos a seguir:

1, 3, 6, 9, 10, 11, 15, 16, 19, 20, 22, 26, 28, 31, 33, 34.

Observe:

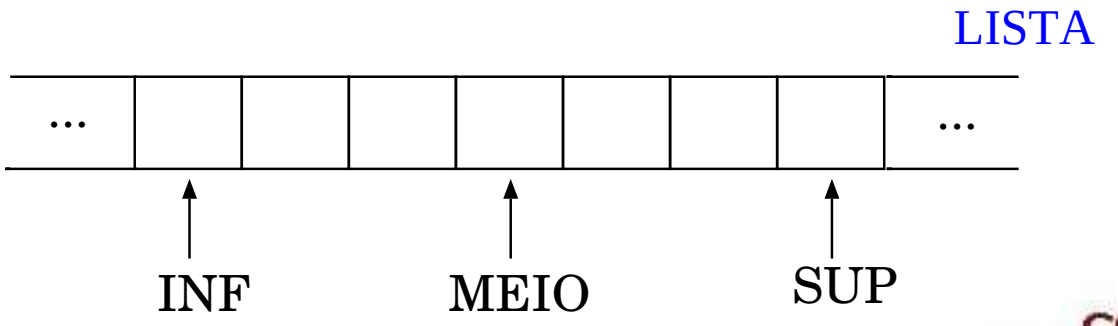
elementos "possíveis"	elemento comparado	elementos descartados
1 até 34	$x = 16$? (não)	1 até 16
19 até 34	$x = 26$? (não)	26 até 34
19 até 22	$x = 20$? (não)	20 até 22
19	$x = 19$? (SIM)	

Em cada passo, o elemento escolhido para a comparação (2ª coluna) está no meio da lista formada pelos elementos da 1ª coluna.

Algoritmo de Busca Binária

Variáveis:

- INF e SUP: Variáveis que marcam o início e o fim da lista formada pelos elementos sendo atualmente considerados.
- MEIO: Variável que marca a posição do elemento a ser comparado.



Algoritmo de Busca Binária

Descrição:

Algoritmo: Busca Binária no vetor V, ordenado, com n elementos.

```
função BUSCA-BIN (x)
    INF := 1;
    SUP := n;
    BUSCA-BIN := 0;
    enquanto INF ≤ SUP faça
        MEIO := ⌊ INF+SUP) / 2 ⌋
        se  V[MEIO] = x
        então  BUSCA-BIN := MEIO      % elemento encontrado
               INF := SUP + 1         % para forçar o término
        senão  se  V[MEIO] <  x
               então  INF := MEIO + 1  % metade posterior
               senão  SUP := MEIO - 1  % metade anterior
```

Exercício

Quantas comparações a Busca Binária efetua para buscar os elementos 14, 16 e 31 na lista abaixo?

- 1, 3, 6, 9, 10, 11, 15, 16, 19, 20, 22, 26, 28, 31, 33, 34

Tempo : 5 minutos

Solução

Para $x = 14$, a Busca Binária compara x com os elementos 16, 9, 11, 15.

Portanto, são efetuadas 4 comparações.

Solução

- Para $x = 14$, a Busca Binária compara x com os elementos 16, 9, 11, 15.
  Portanto, são efetuadas 4 comparações.

- Para $x = 16$, a Busca Binária compara x apenas com o elemento 16.
  Portanto, é efetuada 1 comparação.

Solução

- Para $x = 14$, a Busca Binária compara x com os elementos 16, 9, 11, 15.
  Portanto, são efetuadas 4 comparações.

- Para $x = 16$, a Busca Binária compara x apenas com o elemento 16.
  Portanto, é efetuada 1 comparação.

- Para $x = 31$, a Busca Binária compara x com os elementos 16, 26 e 31.
  Portanto, são efetuadas 3 comparações.

Complexidade de Pior Caso da Busca Binária

- O pior caso ocorre quando o elemento procurado é o último que resta na lista de elementos possíveis, ou quando o elemento procurado não está na lista.

- Observe que, em qualquer uma destas duas situações, o pior caso ocorre quando a busca prossegue até que a lista se resuma a um único elemento.

Complexidade de Pior Caso da Busca Binária

Podemos calcular o número de iterações que a busca binária realiza no pior caso (até que a lista tenha um único elemento):

- 1ª iteração — n elementos na lista
- 2ª iteração — $\lfloor n/2 \rfloor$ elementos na lista
- 3ª iteração — $\lfloor \lfloor n/2 \rfloor / 2 \rfloor$ elementos na lista
- ⋮
- última iteração — 1 elemento na lista

Portanto, o número de iterações é no máximo $1 + \lfloor \log_2 n \rfloor$

Como cada iteração consome um tempo constante, concluímos que a complexidade de pior caso da busca binária é $\theta(\log_2 n)$.