

## Tiago Boeing

Análise de Algoritmos

01 de abril de 2020

Foram criadas três listas contendo os elementos em ordem crescente. Com 11, 21 e 42 itens respectivamente. Foi utilizada a estratégia de busca do elemento e movê-lo para posição desejada, assim é possível uma abordagem mais dinâmica e evita-se fixar os índices no código.

**LongList possui 15324799 elementos (para conseguir medir o tempo de execução, de acordo com recursos de processamento)**

Seguindo o algoritmo fornecido, estamos incrementando o contador apenas no pior caso, quando o elemento analisado na posição do vetor não corresponde ao elemento buscado.

### Buscar elemento e retornar na segunda posição

Sem utilizar delay

| Lista    | Elemento buscado | Iterações (else) | Iterações (while) | Tempo (segs) |
|----------|------------------|------------------|-------------------|--------------|
| Lista 1  | 5                | 5                | 5                 | 0.0          |
| Lista 2  | 11               | 11               | 11                | 0.0          |
| Lista 3  | 21               | 21               | 21                | 0.0          |
| LongList | 100000           | 100000           | 100000            | 0.001        |

Utilizando delay de 2 segundos

| Lista    | Elemento buscado | Iterações (else) | Iterações (while) | Tempo (segs) |
|----------|------------------|------------------|-------------------|--------------|
| Lista 1  | 5                | 5                | 5                 | 2.0          |
| LongList | 10000            | 10000            | 10000             | 2.008        |

### Buscar e retornar elemento na posição mediana

Sem utilizar delay.

| Lista    | Elemento buscado | Iterações (else) | Iterações (while) | Tempo (segs) |
|----------|------------------|------------------|-------------------|--------------|
| Lista 1  | 5                | 5                | 5                 | 0.0          |
| Lista 2  | 11               | 11               | 11                | 0.0          |
| Lista 3  | 21               | 21               | 21                | 0.0          |
| LongList | 10000            | 10000            | 10000             | 0.01         |

Utilizando delay de 2 segundos

| Lista    | Elemento buscado | Iterações (else) | Iterações (while) | Tempo (segs) |
|----------|------------------|------------------|-------------------|--------------|
| Lista 1  | 5                | 5                | 5                 | 2.0          |
| LongList | 10000            | 10000            | 10000             | 2.01         |

### Busca quando elemento não existe

Para verificar que o elemento não existe é necessário varrer todo o vetor. As iterações serão correspondentes ao tamanho do vetor (length).

Sem utilizar delay.

| Lista    | Elemento buscado | Iterações (else) | Iterações (while) | Tempo (segs) |
|----------|------------------|------------------|-------------------|--------------|
| Lista 1  | 50               | 11               | 11                | 0.0          |
| Lista 2  | 50               | 21               | 21                | 0.0          |
| Lista 3  | 50               | 42               | 42                | 0.0          |
| LongList | 10000            | 15324798         | 15324799          | 0.016        |

Utilizando delay de 2 segundos

| <b>Lista</b> | <b>Elemento buscado</b> | <b>Iterações (else)</b> | <b>Iterações (while)</b> | <b>Tempo (segs)</b> |
|--------------|-------------------------|-------------------------|--------------------------|---------------------|
| Lista 1      | 5                       | 11                      | 11                       | 2.008               |
| LongList     | 10000                   | 15324798                | 15324799                 | 2.019               |

O elemento buscado será sempre o índice do vetor (desde que o vetor esteja ordenado de forma crescente). A quantidade de iterações será a mesma, já que temos que iterar a lista X vezes, onde x é o elemento buscado.

Os algoritmos são lineares, quanto maior a quantidade de elementos na lista, maior será a quantidade de iterações e consequentemente o tempo para execução. No caso da busca quando não encontra valores correspondentes (este é o pior caso). A quantidade de iterações será o tamanho do vetor.

Nas buscas onde há resultado a quantidade de iterações será a posição em que o elemento se encontra no array. Embora retornamos o elemento buscado na posição 2 ou mediana, temos que executar as iterações até que o elemento seja localizado, para só depois movê-lo (estratégia utilizada) no vetor em que se encontra ou adicioná-lo em um novo.