**UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC**

**Augusto Savi**

**Arthur Hassan Souki**

**Arthur Neto Bem**

**Relatório Busca Sequencial e Binária**

**Criciúma**

**2019**

**SUMÁRIO**

[1 INTRODUÇÃO](#__RefHeading___Toc488856097) 04

[2](#__RefHeading___Toc488856098) Busca Sequencial05

2.1 [Calculo de Complexidade de Tempo](#__RefHeading___Toc488856100) 05

2.2SIMULAÇÕES [de Tempo](#__RefHeading___Toc488856100) 06

[3](#__RefHeading___Toc488856102) BUSCA BINARIA07

[3.1. CALCULO DE COMPLEXIDADE DE TEMPO](#__RefHeading___Toc488856103) 07

[3.2 SIMULAÇÕES DE TEMPO](#__RefHeading___Toc488856105) 08

[REFERÊNCIAS](#__RefHeading___Toc488856110) 09

# 1 INTRODUÇÃO

O presente trabalho tem como objetivos explicar o funcionamento e os cálculos de complexidades dos métodos de busca sequencial e binaria, mas também mostrar simulações de tempo e como desenvolver esses métodos na linguagem python

# 2 Busca Sequencial

O termo busca sequencial (ou busca linear) e usado para expressar um tipo de pesquisa em listas ou vetores de modo sequencial, verificando elemento por elemento até encontrar a chave.

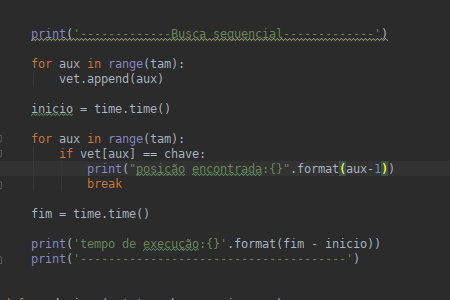
**2.1 Calculo de Complexidade de Tempo**

Esse método possui três casos, no melhor caso o o elemento a ser encontrado está na primeira posição, no pior caso, o elemento a ser encontrado está na ultima posição e são feitas “N” comparações (“N” é o numero total de elementos). No caso médio, o elemento é encontrado no meio do vetor ou da lista.

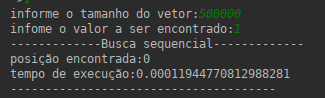
Quanto ao tempo, o algoritmo de busca sequencial usa o tempo linear, ou tempo O(n), isto significa que para quanto maior a entrada de dados, a execução delas aumenta linearmente com o tamanho da entrada, mas deve ser levado em conta os casos citados anteriormente.

**2.2 Simulações de Tempo**

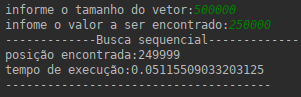
Utilizando um código simples desenvolvido em python, será demonstrado os 3 casos de uma busca em uma lista ordenada de 500 mil posições.



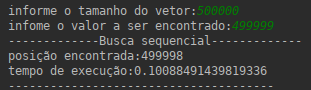
No primeiro caso sera feita uma busca pelo melhor caso que é elemento que se encontra na primeira posição da lista.



No segundo caso será feita uma busca no meio do vetor, que seria o caso médio



No terceiro caso será realizado o pior caso, aonde o elemento será encontrado somente na ultima posição do vetor.



# 3 Busca Binaria

A busca binaria é um algoritmo que parte do pressuposto que o vetor está ordenado e realiza divisões sucessivas no espaço de busca comparando o elemento buscado(chave) com o elemento que se encontra no meio do vetor. Se o elemento do meio vier antes da chave, então a busca continua na metade posterior do vetor, e se o elemento do meio vier depois da chave, a pesquisa continuará na metade anterior do vetor.

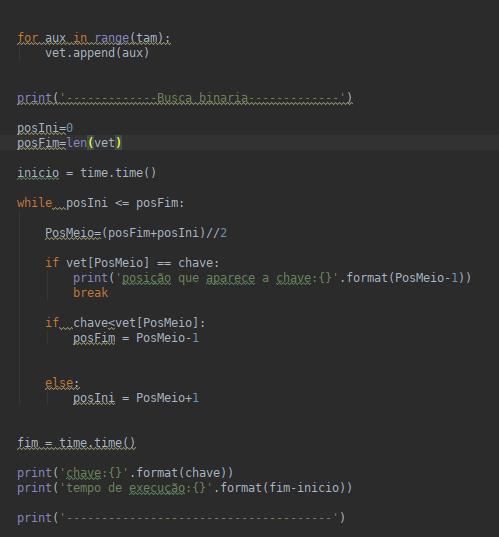
## 3.1 **Calculo de complexidade de tempo**

A complexidade da busca binaria é de ordem O (log n), onde n é o tamanho do vetor, esse método de busca apresenta-se mais eficiente que a busca sequencial.

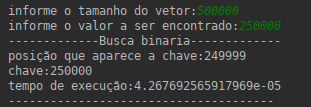
A complexidade de caso médio em uma busca dá-se por O(log n), no melhor caso O(1) e no pior caso O(log n).

### 3.1.1 Simulações de Tempo

Utilizando um código já desenvolvido em aulas anteriores será apresentado os 3 casos de complexidade:

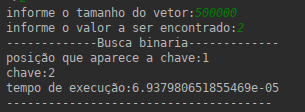


No primeiro caso será feita uma busca pelo melhor caso que é elemento que se encontra no meio do vetor na primeira busca



E como a complexidade de caso médio e de pior caso é o mesmo,

vou mostrar uma de pior caso, aonde a chave a ser encontrada está na ultima busca feita pelo código.



REFERÊNCIAS

Busca linear. Wikipédia . Disponível em: <https://pt.wikipedia.org/wiki/Busca\_linear>. Acesso em: 17 abril de 2019.

Grande-O. Wikipédia . Disponível em: <https://pt.wikipedia.org/wiki/Grande-O#M%C3%BAltiplos\_usos> . Acesso em: 17 abril de 2019 .

Complexidade de tempo. Wikipédia. Disponível em :<https://pt.wikipedia.org/wiki/Complexidade\_de\_tempo#Tempo\_linear> . Acesso em :17 abril de 2019.

Pesquisa binária. Wikipédia. Disponível em :<https://pt.wikipedia.org/wiki/Pesquisa\_bin%C3%A1ria> . Acesso em: 17 abril de 2019