UNIVERSIDADE PAULISTA (UNIP)

CIÊNCIA DA COMPUTAÇÃO

**ATIVIDADES PRÁTICAS SUPERVISIONADAS**

**DESENVOLVIMENTO DE SISTEMA PARA ANÁLISE DE PERFORMANCE DE ALGORITMOS DE ORDENAÇÃO DE DADOS**

ANDREICK GOMES MAIA - F044289

BEATRIZ ALMEIDA DE LIMA SOUZA - N459JB4

MAICOLN VINICIUS DOS S. SILVA - F0551I7

MATEUS CARVALHO DA SILVA - N508620

SÃO PAULO

2020

**Sumário**

[Índice 3](#_Toc53685153)

[Objetivo 3](#_Toc53685154)

[Introdução 3](#_Toc53685155)

[Métodos de Ordenação 3](#_Toc53685156)

[Método QuickSort 3](#_Toc53685157)

[Método SelectionSort 3](#_Toc53685158)

[Método InsertiomSort 3](#_Toc53685159)

[Desenvolvimento do Sistema 3](#_Toc53685160)

[Geração e Obtenção de Dados 3](#_Toc53685161)

[Ordenação de Dados 3](#_Toc53685162)

[Comparativo 3](#_Toc53685163)

[Resultados 4](#_Toc53685164)

[Conclusão 4](#_Toc53685165)

[Referências Bibliográficas 4](#_Toc53685166)

[Código Fonte 4](#_Toc53685167)

[Fichas 4](#_Toc53685168)

# Objetivo

O geoprocessamento de imagens da floresta amazônica permite a fiscalização de ações de crimes ambientais. Os satélites geram cerca de 100 mil imagens de toda a região a cada 24 horas, essas imagens são armazenadas o catalogadas.

O grupo deverá selecionar três ou mais técnicas e desenvolver um sistema computacional completo que obtenha os dados catalogados das imagens capturadas dos satélites, efetue a ordenação e compare os desempenhos entre eles

A unidade de medida para efeito de comparação deverá ser o tempo total de ordenação.

# Introdução

Desde os primórdios, com o surgimento da matemática, a organização de dados se faz necessária para a resolução de problemas. Como preza a estratégia algorítmica: “Primeiro coloque os números em ordem. Depois decidimos o que fazer.”

Na ciência da computação, uma Estrutura de Dados é um meio de armazenar e organizar dados em um computador, de modo que possam ser usados eficientemente, facilitando sua busca e modificação. Na computação existe uma série de algoritmos que utilizam diferentes técnicas de ordenação para organizar um conjunto de dados, eles são conhecidos como Métodos de Ordenação ou Algoritmos de Ordenação. Quando estes dados estão organizados de forma coerente, caracterizam uma estrutura de dados.

Algoritmos de Ordenação permitem colocar os elementos de uma dada sequência em uma certa ordem. Ou seja, efetuam sua ordenação completa ou parcial. O objetivo da ordenação é facilitar a recuperação dos dados de uma lista. Os mais populares algoritmos de ordenação são: *Insertion sort, Selection sort, Bubble sort, Comb sort, Quick sort, Merge sort, Heap sort* e *Shell sort*.

Neste trabalho, serão abordados os algoritmos *Selection Sort, Quick sort e Insertion sort,* explicando o funcionamento e comparando a eficiência de cada um deles.

# Métodos de Ordenação

Os métodos de ordenação podem ser classificados em métodos de ordenação interna ou ordenação externa, onde a diferença está, basicamente, no armazenamento e no acesso aos registros.

**Ordenação Interna:** os elementos a serem ordenados cabem na memória principal e qualquer registro pode ser imediatamente acessado.

**Ordenação Externa:** os elementos a serem ordenados não cabem na memória principal e os registros são acessados sequencialmente ou em grandes blocos.

Nos algoritmos de ordenação as medidas de complexidade relevantes são:

* Número de comparações C(n) entre chaves.
* Número de movimentações M(n) dos registros dos vetores.
* Onde n é o número de registros.

Dentro da ordenação interna temos os Métodos Simples e os Métodos Eficientes.

**Métodos simples:** são adequados para pequenos vetores, são programas pequenos e fáceis de entender. Possuem complexidade C(n) = O(n²), ou seja, requerem O(n²) comparações.

Exemplos*: Insertion Sort, Selection Sort, Bubble Sort, Comb Sort*.

**Métodos Eficientes:** são adequados para vetores maiores e mais complexos. Requerem menos comparações, de complexidade C(n) = O(n log n).

Exemplos: *Shellsort, Heapsort, Mergesort, Quicksort*.

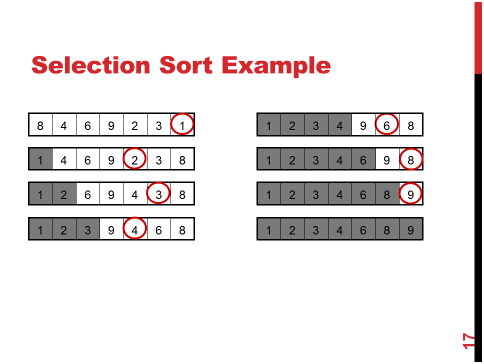
A seguir, será explicado o funcionamento os algoritmos de ordenação interna: *Selection Sort, Quick sort* e *Insertion sort*.

## 

## Método *Selection Sort*

Na ordenação por seleção ou *Selection Sort*, é escolhido um número a partir do primeiro e comparado com os números a partir da sua direita. Quando encontrado um número menor, o número escolhido ocupa sua posição. Este número encontrado será o próximo número escolhido. Caso não for encontrado nenhum número menor que este escolhido, ele é colocado na posição do primeiro número escolhido, e o próximo número à sua direita vai ser o escolhido para fazer as comparações. É repetido esse processo até que a lista esteja ordenada.

O *Selection Sort* é notado por sua simplicidade e tem vantagens de desempenho sobre algoritmos mais complicados em certas situações, particularmente onde a memória auxiliar é limitada.

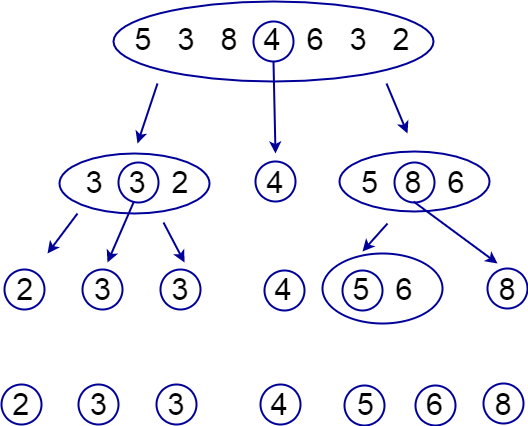


1. A partir do primeiro elemento, pesquisamos o menor elemento da matriz, e o substituímos pelo elemento na primeira posição.
2. A partir do primeiro elemento, pesquisamos o menor elemento da matriz, e o substituímos pelo elemento na primeira posição.
3. Em seguida vamos para a segunda posição e percorremos a lista procurando o menor elemento presente no sub-array, partindo do índice atual até o ultimo, inserindo o segundo menor valor na segunda posição.
4. Isso se repete até tudo estar em ordem.

Para todos os casos possui complexidade C(n) = O(n²) e não é um algoritmo estável.

## Método QuickSort

O Algoritmo Quicksort, criado por C. A. R. Hoare em 1960, é o método de ordenação interna mais rápido que se conhece para uma ampla variedade de situações. É um algoritmo de comparação que emprega a estratégia de “divisão e conquista”. A ideia básica é dividir o problema de ordenar um conjunto com n itens em dois problemas menores. Os problemas menores são ordenados independentemente e os resultados são combinados para produzir a solução final.



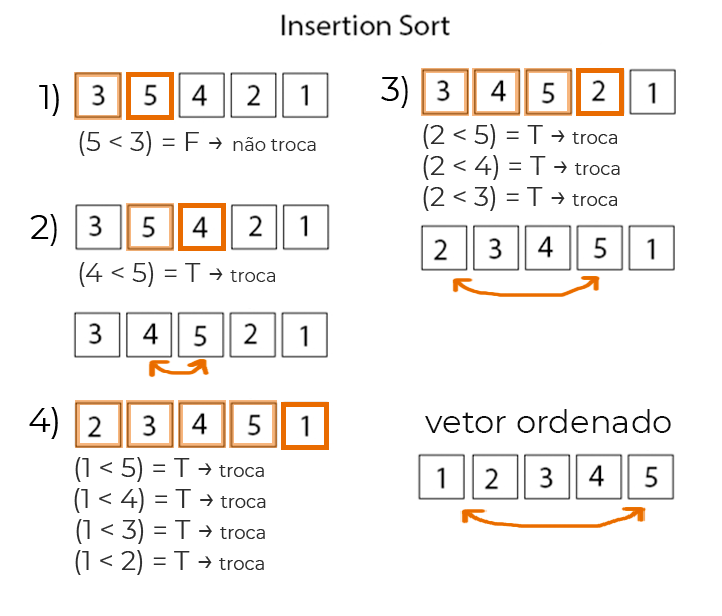
Basicamente a operação do algoritmo pode ser resumida na seguinte estratégia: divide sua lista de entrada em duas sub-listas a partir de um pivô, para em seguida realizar o mesmo procedimento nas duas listas menores até uma lista unitária.

A principal desvantagem deste método é que ele possui uma implementação difícil e delicada, um pequeno engano pode gerar efeitos inesperados para determinadas entradas de dados.

Possui complexidade C(n) = O(n²) no pior caso e C(n) = O(n log n) no melhor e médio caso. Não é um algoritmo estável.

## Método InsertionSort

O Insertion sort é um algoritmo simples e eficiente quando aplicado em pequenas listas. Neste algoritmo a lista é percorrida da esquerda para a direita, à medida que avança vai deixando os elementos mais à esquerda ordenados.

O funcionamento do algoritmo é bem simples: consiste em cada passo a partir do segundo elemento selecionar o próximo item da sequência e colocá-lo no local apropriado de acordo com o critério de ordenação. É semelhante à forma que as pessoas usam para ordenar cartas em um jogo de baralho como o pôquer.

Possui complexidade C(n) = O(n) no melhor caso e C(n) = O(n²) no caso médio e pior caso. É considerado um método de ordenação estável, ou seja, a ordem relativa dos itens iguais não se altera durante a ordenação.

# Desenvolvimento do Sistema

## Geração e Obtenção de Dados

## Ordenação de Dados

processo de ordenação de dados, listagem dos valores antes e depois da ordenação

## Comparativo

apresentação dos resultados comparativos de performance.

# Resultados

6pgs demonstrar a eficiência dos algoritmos de ordenação em diversos cenários possíveis (tamanhos variados de valores de entrada, dados ordenados, semi-ordenados ou aleatórios, etc ...). É fundamental que os resultados sejam apresentados por meio de tabelas e gráficos, e os resultados obtidos sejam discutidos com o intuito de identificar as vantagens e desvantagens de cada algoritmo, em determinado cenário;

# Conclusão

1pg considerações do grupo a respeito das técnicas escolhidas, bem como sobre os resultados obtidos;

# Referências Bibliográficas

2pgs

# Código Fonte

# Fichas