**UNIVERSIDADE LUTERANA DO BRASIL**

**EDUARDO ARAÚJO DOS SANTOS**

**ORDENAÇÃO DE DADOS**

Charqueadas/RS

2020

Veremos neste trabalho estudos sobre a ordenação de dados, comparativos entre os métodos de classificações tais como: Ordenação por troca, por inserção e por seleção. Cada método terá seu algoritmo/código, e serão mostradas suas vantagens e desvantagens e qual o mais eficiente.

1. **ORDENAÇÃO POR TROCA (Bubble Sort)**

Este método faz com que haja uma varredura no arranjo do primeiro ao ultimo elemento e vai fazendo comparações dos valores dois a dois e se o primeiro elemento for maior que o segundo, é realizado uma troca de lugar entre eles no arranjo. Então esta varredura é realizada várias vezes, de forma que se existir um elemento fora da ordem crescente a sua posição vai sendo trocada até que ele fique posicionado corretamente.

De vantagem ele tem e é um método fácil de se utilizar e é o procedimento mais simples de se fazer uma ordenação e sua desvantagem é ser um dos mais lentos, pois pra fazer toda a ordenação ele tem que varrer o arranjo varias vezes.

**void bubblesort(int num, int \*vetor) {**

**int i, j;**

**for (i = num - 1; i >= 1; i--) {**

**int troca = 0;**

**for (j = 0; j < i; j++)**

**if (vetor[j] > vetor[j + 1]) {**

**int aux = vetor[j];**

**vetor[j] = vetor[j + 1];**

**vetor[j + 1] = aux;**

**troca = 1;**

**}**

**if (troca == 0)**

**return;**

**}**

**}**

*(Método criado em C referente ao artigo ANÁLISE COMPARATIVA DE MÉTODOS DE ORDENAÇÃO)*

1. **ORDENAÇÃO POR INSERÇÃO (Insersion Sort)**

Este método faz com os n dados sejam divididos em dois arranjos: Um arranjo ordenado e outro que vai ser ordenado. No inicio, o primeiro arranjo é formado por apenas um elemento, que pode ser considerado como já ordenado e no segundo arranjo é formado pelos n-1 elementos restantes. Partindo disto o processo se desenvolve e n-1 iterações, de modo que em cada uma delas um elemento do arranjo não ordenado acabe sendo transferido para o arranjo ordenado e sendo inserido em sua posição correta em relação àqueles que já estão no arranjo.

De vantagem ele é o que faz o menor número de comparações e de desvantagem é a excessão de números de trocas que ele faz e ordenando como se fosse cartas de baralhos.

**void insertsort(int num, int \*vetor){**

**int i, j, z;**

**for (i=0; i<num; i++){**

**z=vetor[i];**

**for (j=i-1; (j>=0) && (z<vetor[j]); j--)**

**vetor[j+1]=vetor[j];**

**vetor[j+1]=z;**

**}**

**}**

*(Método criado em C referente ao artigo ANÁLISE COMPARATIVA DE MÉTODOS DE ORDENAÇÃO)*

1. **ORDENAÇÃO POR SELEÇÃO (Selection Sort)**

Este método de ordenação por seleção é caracterizado por selecionar, a cada repetição, o menor ou o maior elemento do arranjo, ele faz com que esse elemento (menor ou maior) seja trocado com o primeiro elemento do arranjo. A seguir, repete-se o processo, desconsiderando-se a parte do arranjo que já está ordenada. A ordenação se encerra quando restar somente um elemento não selecionado, que vai estar na sua posição correta no arranjo.

De vantagem de seu uso ocorre quando se trabalha com componentes em que, quanto mais se escreve, ou reescreve, mais se desgasta, e, consequentemente, perdem sua eficiência, como é o caso das memórias EEPROM e FLASH e desvantagem consome muito tempo de leitura e menos de escrita.

**void selectsort(int num, int \*vetor) {**

**int i, j, z, p;**

**int tam = num;**

**tam--;**

**for (i = 0; i < tam; i++) {**

**p = i;**

**for (j = i + 1; j <= tam; j++)**

**if (vetor[j] < vetor[p])**

**p = j;**

**if (p != i) {**

**z = vetor[p];**

**vetor[p] = vetor[i];**

**vetor[i] = z;**

**}**

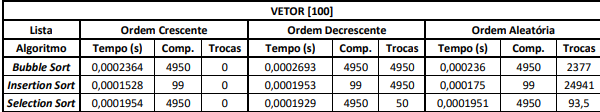
**}**

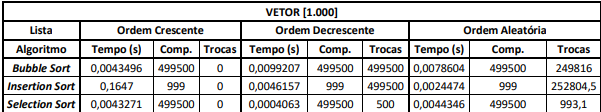
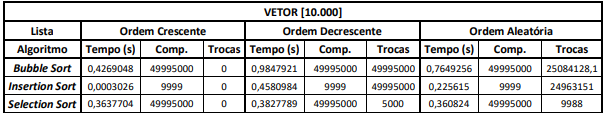
**}**

*(Método criado em C referente ao artigo ANÁLISE COMPARATIVA DE MÉTODOS DE ORDENAÇÃO)*

1. **TESTES**

No artigo Algoritmos de Ordenação: Um Estudo Comparativo foram feitos 3 testes a qual na primeira tentativa, foi feito um teste de desempenho com vetor de 100 elementos, na segunda tentativa 1000 elementos e na terceira 10.000 elementos sendo utilizando os 3 métodos citados ao longo deste trabalho sendo:



1. **CONCLUSÃO**

O Bubble Sort, apesar de ser o de mais fácil implementação, não apresenta resultados satisfatórios, principalmente no número de comparações.

O Selection Sort torna-se útil em estruturas lineares similares ao do Insertion Sort, porém, com o número de elementos consideravelmente maior

O insertion Sort de todos os testes foi o que possuiu melhor desempenho. Ele é útil para estruturas lineares pequenas, geralmente entre 8 e 20 elementos, sendo amplamente utilizados em sequências de 10 elementos, tendo ainda, listas ordenadas de forma decrescente como pior caso, listas em ordem crescente como o melhor caso e, as demais ordens como sendo casos medianos. Sendo assim de todos os 3 métodos esse seria o melhor.

1. **REFERÊNCIA BIBLIOGRÁFICA**

<https://www.researchgate.net/publication/308900585_Analise_comparativa_de_metodos_de_ordenacao> Partes do texto citadas por (IVERSON 1962) (CELES, CERQUEIRA e RANGEL 2004) (SILVA 2007), (SIQUEIRA 2005), (CELES, CERQUEIRA e RANGEL 2004).

[file:///C:/Users/e-a-d\_000/Downloads/7082-Texto%20do%20artigo-33807-2-10-20180903%20(1).pdf](file:///C:\Users\e-a-d_000\Downloads\7082-Texto%20do%20artigo-33807-2-10-20180903%20(1).pdf)(Algoritmos de Ordenação: Um Estudo Comparativo Jackson É. G. Souza¹, João V. G. Ricarte¹, Náthalee C. A. Lima²)Partes do texto citadas por (SILVA, 2010)