

ANTONIO MENEGHETTI FACULDADE



MÉTODOS DE ORDENAÇÃO

TÓPICOS AVANÇADOS EM PROGRAMAÇÃO

André Redin Cella

Professor: Fernando Luís Herman

RECANTO MAESTRO

2016

1- INTRODUÇÃO

Algoritmos de ordenação são muito usados na resolução de problemas computacionais. Eles servem para ordenar e organizar uma lista de números ou palavras de acordo com a sua necessidade. As linguagens de programação já possuem métodos de ordenação, mas é bom saber como funcionam os algoritmos, pois há casos de problemas em que o algoritmo de ordenação genérico não resolve, às vezes é necessário modificá-lo.

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 estudados os algoritmos Insertion Sort, Selection Sort, Quick Sort, Heap Sort e Shell Sort.

2- INSERTION SORT

Ele é o método de ordenação mais rápido entre os outros métodos considerados básicos. A principal característica deste método consiste em ordenar o arranjo utilizando um sub-arranjo ordenado localizado em seu início, a cada passo, acrescenta a este sub-arranjo mais um elemento, até que atinja o último elemento do arranjo fazendo assim com que ele se torne ordenado.

O número máximo ocorre quando os itens estão originalmente na ordem reversa. É um método utilizado para quando o arquivo está quase ordenado.

3- SELECTION SORT

Este algoritmo é baseado em se passar sempre o menor valor do vetor para a primeira posição, depois o segundo menor valor para a segunda posição e assim sucessivamente, até os últimos dois elementos. Ele consiste em encontrar a menor chave por pesquisa sequencial. Encontrando a menor chave, essa é permutada com a que ocupa a posição inicial do vetor, que fica então reduzido a um elemento.

4- QUICK SORT

Esse algoritmo, usa o paradigma da divisão e conquista para resolver o problema, ele é muito rápido em média, mas lento no pior caso.

Porém, sua implementação é muito delicada e difícil, já que um pequeno engano levar a efeitos inesperados para algumas entradas de dados. Ele é um método considerável não estável.

5- HEAP SORT

Esse algoritmo é muito eficiente, mas consome tempo proporcional, mesmo no pior caso. A base do algoritmo é uma fila de propriedades muito eficiente.

Ele utiliza uma estrutura de dados chamada heap binário para ordenar os elementos a medida que os insere na estrutura. Assim, ao final das inserções, os elementos podem ser sucessivamente removidos da raiz da heap, na ordem desejada.

O comportamento do Heap Sort é sempre o mesmo, independentemente de qual seja a entrada, porém seu anel interno é bastante complexo e é considerado um algoritmo não estável.

6- SHELL SORT

É uma ótima opção para arquivos de tamanho moderado, sua implementação é simples e requer uma quantidade de código pequena. O tempo de execução do algoritmo é sensível à ordem inicial do arquivo, e o método não é estável.

Ele é um algoritmo não muito conhecido, ninguém sabe da sua eficiência, pois contém alguns problemas matemáticos muito difíceis, como por exemplo, a sequência de incrementos, a única coisa que se sabe é que cada incremento não deve ser múltiplo do anterior.

7- UTILIZAÇÃO DO PROGRAMA

- 1- Irá aparecer um menu, onde o usuário deverá escolher qual opção desejar, veja na imagem abaixo.




```

C:\Users\André Redin\Desktop\Trabalho TAP.exe
|***** Metodos de Ordenacao: *****|
<1> - Insertion Sort
<2> - Selection Sort
<3> - Quick Sort
<4> - Heap Sort
<5> - Shell Sort
<6> - Sair
Opcao:

```

- 2- Digite qual opção quiser e aperte a tecla ENTER, igual imagem;

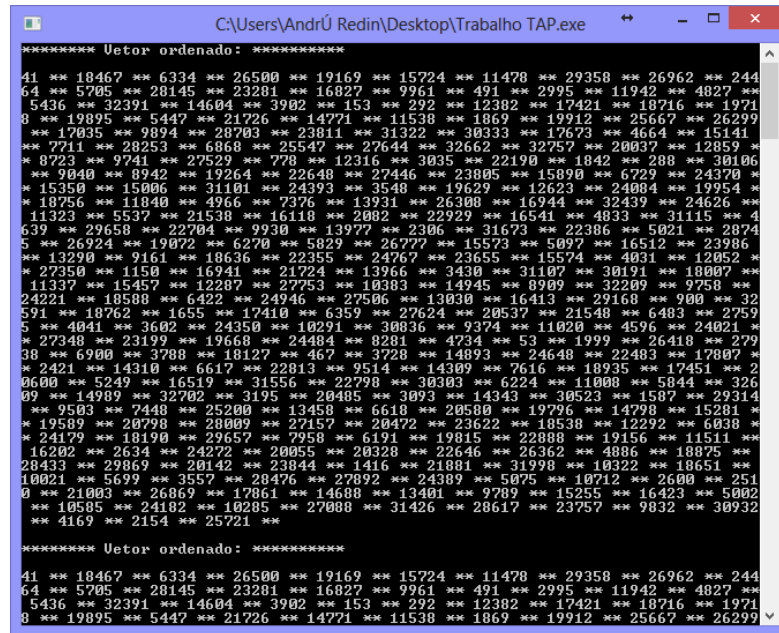


```

C:\Users\André Redin\Desktop\Trabalho TAP.exe
|***** Metodos de Ordenacao: *****|
<1> - Insertion Sort
<2> - Selection Sort
<3> - Quick Sort
<4> - Heap Sort
<5> - Shell Sort
<6> - Sair
Opcao: 2

```

- 3- Pronto, no momento que você apertar a tecla ENTER, irá gerar os os 300 vetores.



```
***** Vetor ordenado: *****
41 ** 18467 ** 6334 ** 26500 ** 19169 ** 15724 ** 11478 ** 29358 ** 26962 ** 244
64 ** 5705 ** 28145 ** 23281 ** 16827 ** 9961 ** 491 ** 2995 ** 11942 ** 4827 **
5436 ** 32391 ** 14604 ** 3902 ** 153 ** 292 ** 12382 ** 17421 ** 18716 ** 1971
8 ** 19895 ** 5447 ** 21726 ** 14771 ** 11538 ** 1869 ** 19912 ** 25667 ** 26299
** 17035 ** 9894 ** 28703 ** 23011 ** 31322 ** 30333 ** 17673 ** 4664 ** 15141
** 7711 ** 28253 ** 6868 ** 25547 ** 27644 ** 32662 ** 32757 ** 20037 ** 12859 **
** 8723 ** 9741 ** 27529 ** 778 ** 12316 ** 3035 ** 22190 ** 1842 ** 288 ** 30106
** 9040 ** 8942 ** 19264 ** 22640 ** 27446 ** 23005 ** 15890 ** 6729 ** 24370 **
** 15350 ** 15006 ** 31101 ** 24393 ** 3548 ** 19629 ** 12623 ** 24084 ** 19954 **
** 18756 ** 11840 ** 4966 ** 7376 ** 13931 ** 26308 ** 16944 ** 32439 ** 24626 **
11323 ** 5537 ** 21538 ** 16118 ** 2082 ** 22929 ** 16541 ** 4833 ** 31115 ** 4
639 ** 29650 ** 22704 ** 9930 ** 13977 ** 2306 ** 31673 ** 22386 ** 5021 ** 2074
5 ** 26924 ** 19072 ** 6270 ** 5829 ** 26777 ** 15573 ** 5097 ** 16512 ** 23986
** 13290 ** 9161 ** 18636 ** 22355 ** 24767 ** 23655 ** 15574 ** 4031 ** 12052 **
** 27350 ** 1150 ** 16941 ** 21724 ** 13966 ** 3430 ** 31107 ** 30191 ** 18007 **
11337 ** 15457 ** 12287 ** 27753 ** 10383 ** 14945 ** 8909 ** 32209 ** 9758 **
24221 ** 18588 ** 6422 ** 24946 ** 27506 ** 13030 ** 16413 ** 29168 ** 900 ** 32
591 ** 18762 ** 1655 ** 17410 ** 6359 ** 27624 ** 20537 ** 21548 ** 6403 ** 2759
5 ** 4041 ** 3602 ** 24350 ** 10291 ** 30036 ** 9374 ** 11020 ** 4596 ** 24021 **
27340 ** 23199 ** 19668 ** 24484 ** 8281 ** 4734 ** 53 ** 1999 ** 26410 ** 279
38 ** 6900 ** 3788 ** 18127 ** 467 ** 3728 ** 14893 ** 24648 ** 22483 ** 17807 **
** 2421 ** 14310 ** 6617 ** 22813 ** 9514 ** 14309 ** 7616 ** 18935 ** 17451 ** 2
0000 ** 5249 ** 16519 ** 31556 ** 22798 ** 30303 ** 6224 ** 11000 ** 5844 ** 326
09 ** 14989 ** 32702 ** 3195 ** 20405 ** 3093 ** 14343 ** 30523 ** 1587 ** 29314
** 9503 ** 7448 ** 25200 ** 13458 ** 6618 ** 20500 ** 19796 ** 14798 ** 15281 **
** 19589 ** 20798 ** 28009 ** 27157 ** 20472 ** 23622 ** 18530 ** 12292 ** 6038 **
** 24170 ** 18190 ** 29657 ** 7950 ** 6191 ** 19815 ** 22000 ** 19156 ** 11511 **
16202 ** 2634 ** 24272 ** 20055 ** 20328 ** 22646 ** 26362 ** 4886 ** 18875 **
28433 ** 29869 ** 20142 ** 23844 ** 1416 ** 21881 ** 31998 ** 10322 ** 18651 **
10021 ** 5699 ** 3557 ** 28476 ** 27892 ** 24389 ** 5075 ** 10712 ** 2600 ** 251
0 ** 21003 ** 26869 ** 17861 ** 14688 ** 13401 ** 9709 ** 15255 ** 16423 ** 5002
** 10585 ** 24182 ** 10285 ** 27088 ** 31426 ** 28617 ** 23757 ** 9832 ** 30932
** 4169 ** 2154 ** 25721 **
***** Vetor ordenado: *****
41 ** 18467 ** 6334 ** 26500 ** 19169 ** 15724 ** 11478 ** 29358 ** 26962 ** 244
64 ** 5705 ** 28145 ** 23281 ** 16827 ** 9961 ** 491 ** 2995 ** 11942 ** 4827 **
5436 ** 32391 ** 14604 ** 3902 ** 153 ** 292 ** 12382 ** 17421 ** 18716 ** 1971
8 ** 19895 ** 5447 ** 21726 ** 14771 ** 11538 ** 1869 ** 19912 ** 25667 ** 26299
```

8- REFERENCIAS

[1]DEVMEDIA, Disponível em: < <http://www.devmedia.com.br/algoritmos-de-ordenacao-analise-e-comparacao/28261> > . Acesso em 14 de abril de 2016.

[2]UNICAMP, Disponível em: < <http://www.ft.unicamp.br/liag/siteEd/definicao/insertion-sort.php> > Acesso em 17 de abril de 2016.

[3]DCC, Disponível em: < <http://homepages.dcc.ufmg.br/~cunha/teaching/20121/aeds2/sortingcmp.pdf> > . Acesso em 19 de abril de 2016.