

# Escola Superior Dom Helder Câmara

Escola de Engenharia

Disciplina: Algoritmo e Estruturas de Dados II

Professor: Diego Silva Caldeira Rocha

Lista 6, 7 – Métodos de ordenação

Para cada problema proposto postar as soluções no Moodle. Para isto, compacte em único  
arquivo ou conjunto das soluções (os arquivos com extensão .JAVA e docx, pdf..).

1)Implemente os algoritmos em JAVA:B ubblleSort, SelectionSort , InsertSort, Mergesort, QuickSort ,ShellSort e HeapSort. Para o tipo **DOUBLE**.

O código a seguir calcula o tempo gasto para executar uma determinada função. Modifique-o para que possa mensurar o tempo estimado para a ordenação dos métodos implementados. Além disso, contém uma função de geração de valores aleatórios e decrescente. DICA: não se esqueça de randomizar o vetor antes de inicializar a contagem do tempo para cada algoritmo.

public static void main(String[] args) {

double [] vet= gerarAleatorio(10000, 1, 50000);

long startTime = System.nanoTime();

// Coloque sua função aqui de ordenação

suaFuncao(...);

long endTime = System.nanoTime();

// Calcula o tempo decorrido em milissegundos

long elapsedTimeInMillis = (endTime - startTime) ;

System.out.println("Tempo decorrido: " + elapsedTimeInMillis + " nanosegundos"); }

public static void suaFuncao(...) {

// Implemente sua função aqui

}

// Método para gerar um vetor de valores aleatórios double dentro de um intervalo

public static double[] gerarAleatorio(int tamanho, double minimo, double maximo) {

double[] vetorAleatorio = new double[tamanho];

Random random = new Random();

for (int i = 0; i < tamanho; i++) {

vetorAleatorio[i] = minimo + (maximo - minimo) \* random.nextDouble(); // valores dentro do intervalo

}

return vetorAleatorio;

}

// Método para gerar um vetor decrescente com aleatorização da parte fracionada

public static double[] gerarDecrescenteAleatorio(int tamanho) {

double[] vetorDecrescenteAleatorio = new double[tamanho];

Random random = new Random();

for (int i = 0; i < tamanho; i++) {

double valor = tamanho - i; // gera o valor decrescente

// Aleatoriza a parte fracionada

double parteFracionadaAleatoria = random.nextDouble();

valor += parteFracionadaAleatoria;

vetorDecrescenteAleatorio[i] = valor;

}

return vetorDecrescenteAleatorio;

}

2) Execute os algoritmos e anote os resultados na tabela a seguir. obs: execute o algoritmo em uma mesma máquina.

Tabela 1 Vetor já ordenado

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tamanho do Vetor | BublleSort | SelectionSort | InsertSort | Mergesort | Quicksort | Shellsort | HeapSort |
| 10000 | 6.38322E7 | 4546601.0 | 2.54103E7 | 4345299.0 | 2113799.0 | 7410699.0 | 786300.0 |
| 15000 | 1.19310501E8 | 3407600.0 | 4.29768E7 | 6076001.0 | 3901500..0 | 647900.0 | 1129400.0 |
| 30000 | 4.426464E8 | 2047900.0 | 1.50965699E8 | 5887500.0 | 6298500.0 | 4455500.0 | 2200700.0 |

Tabela 2 Vetor Aleatório

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tamanho do Vetor | BublleSort | SelectionSort | InsertSort | Mergesort | Quicksort | Shellsort | HeapSort |
| 10000 | 772499.0 | 1701.0 | 569499.0 | 739700.0 | 739700.0 | 1536899.0 | 3600.0 |
| 15000 | 3.189032E8 | 4.566274E8 | 1.41590501E8 | 5301700.0 | 2818099.0 | 4794300.0 | 4592900.0 |
| 30000 | 1.4263684E9 | 1.944238E9 | 5.774681E8 | 4713399.0 | 4937700.0 | 3608700.0 | 4257201.0 |

Tabela 3 Vetor Decrescente

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tamanho do Vetor | BublleSort | SelectionSort | InsertSort | Mergesort | Quicksort | Shellsort | HeapSort |
| 10000 | 2.66625E7 | 4.77791E7 | 1.163222E8 | 2395701.0 | 2064800.0 | 3367201.0 | 2842900.0 |
| 15000 | 4.46376E7 | 8.752E7 | 2.256874401E8 | 1302200.0 | 242901.0 | 1230199.0 | 1527900.0 |
| 30000 | 1.577429E8 | 3.402558E8 | 9.239533E8 | 2876800.0 | 406700.0 | 814201.0 | 3213501.0 |

3) Faça uma análise a respeito dos resultados obtidos no seu benchmark das tabela anteriores.

**Informações sobre cópias**

As questões são individuais. Em caso de cópias de trabalho a pontuação será zero para os autores originais e copiadores. Não serão aceitas justificativas como: “Fizemos o trabalho juntos, por isso estão idênticos”.