CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

Departamento de Computação - DECOM

Laboratório de Algoritmos e Estrutura de Dados II

Prática 05 - HeapSort

Thiago Ribeiro Corrêa - Engenharia da Computação

Professor: Otaviano Martins

Belo Horizonte

06 de outubro de 2023

1. Experimentos:

A. Gerar vetores de n elementos ordenados em **ordem crescente**, com n variando de 10.000 até 100.000, com intervalo de 10.000. Ordenar esses vetores utilizando o método HeapSort implementado e computar o número de comparações realizadas.

Número de elementos	Número de comparações
10000	19984
20000	39982
30000	59980
40000	79980
50000	99982
60000	119978
70000	139982
80000	159978
90000	179976
100000	199980

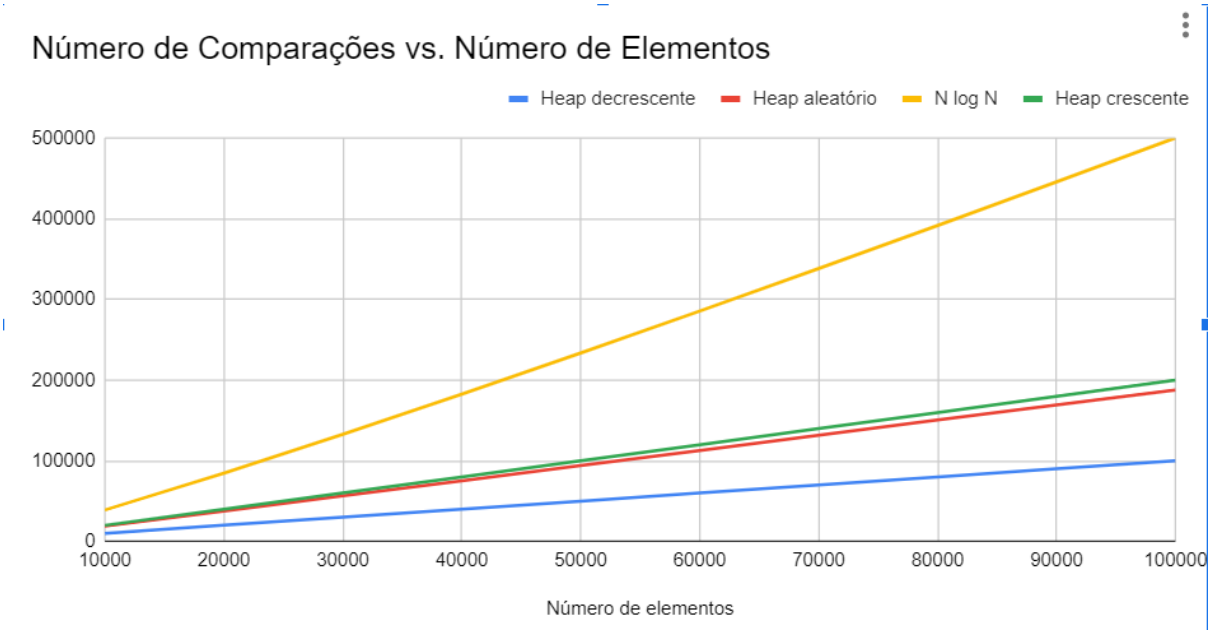
B. Gerar vetores de n elementos ordenados em **ordem decrescente**, com n variando de 10.000 até 100.000, com intervalo de 10.000. Ordenar esses vetores utilizando o método HeapSort implementado e computar o número de comparações realizadas.

Número de elementos	Número de comparações
10000	10000
20000	20000
30000	30000
40000	40000
50000	50000
60000	60000
70000	70000
80000	80000
90000	90000
100000	100000

C. Gerar vetores de n elementos **aleatórios**, com n variando de 10.000 até 100.000, com intervalo de 10.000. Ordenar esses vetores utilizando o método HeapSort implementado e computar o número de comparações realizadas.

Número de elementos	Número de comparações
10000	18782
20000	37498
30000	56484
40000	75190
50000	94036
60000	112914
70000	131586
80000	150582
90000	169372
100000	188040

2. Análise de gráfico



A partir da análise do gráfico acima, o qual analisa o número de comparações necessárias para ordenar um vetor através do método heapsort implementado e o valor de $n \cdot log(n)$, sendo n o número de elementos inseridos, foi possível concluir que a ordenação de um vetor já inserido em ordem decrescente possui custo inferior aos demais. Enquanto a ordenação de um vetor já inserido em ordem crescente possui custo semelhante ao inserido aleatoriamente, ademais, é possível concluir que estes possuem custo aproximadamente igual a $\frac{n \log n}{2}$. Por fim, é possível notar

que, independentemente da ordenação do vetor a ser reordenado pelo Heapsort, o número de comparações aumenta à medida que aumentamos o número de elementos.