



Prática 04 – Implementação em JAVA do TAD Heaps

- **Data de Entrega: 20/04/2018**
- **Deve ser entregue um relatório via moodle:**
 - **Nome, Data de entrega, Número da Prática e Título da Prática;**
 - **Gráfico do tópico D do item 5;**
 - **Explicação do Item 6;**
- **Deve ser entregue, via moodle, projeto contendo o código fonte comentado;**

1) Utilizando o Netbeans, crie um projeto chamado *Prática04*;

2) Implemente a classe **Item**, como especificada abaixo para ser utilizada no T.A.D.;

```
package Item;
public class Item {
    private int chave;
    public Item(int chave) {
        this.chave = chave;
    }
    public int compara(Item it) {
        Item item = it;
        if (this.chave < item.chave)
            return -1;
        else if (this.chave > item.chave)
            return 1;
        return 0;
    }
    public int getChave() {
        return chave;
    }
}
```

3) Implemente uma classe chamada **JHeap** para construir um heap a partir de um dado vetor já preenchido passado por parâmetro no construtor.

4) Implemente uma classe chamada **JHeapSort** para realizar a ordenação de um vetor utilizando a estrutura de **Heap** – método **HeapSort**;

5) Realizar os seguintes 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**;

- 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**;

- 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**;

- d) Fazer um único gráfico de **n** × **número de comparações** levando em consideração as árvores geradas com vetores ordenados em ordem crescente, decrescente e aleatório. Trace um gráfico **n log(n)** para comparar os resultados.

6) Explique o comportamento dos gráficos gerados;