# Exercício 5

## **Quick Sort:**

Algoritmo de ordenação que divide a lista em duas partes de forma contínua e move os items de menor valor para um lado e os outros, de maior valor, para outro. Essa comparação para que os itens sejam alocados de acordo com o seu valor é feita a partir de um valor de referência (pivô). Assim, os itens menores que o pivô são movidos para o lado esquerdo do mesmo, enquanto aqueles maiores ou iguais são organizados à direita. Após o primeiro passo, cada metade repete esse processo continuamente até restar apenas um item no grupo.

**Heap Sort:**

Algoritmo que utilizada a estrutura de dados denominada heap binário que corresponde a uma árvore binária organizada em um vetor. Cada elemento inserido na estrutura é imediatamente ordenado de modo crescente ou decrescente (dependerá da escolha por um max-heap ou um min-heap) e, após as inserções, os elementos são removidos de modo sucessivo da estrutura, mantendo-se a ordem desejada e a estrutura max-heap ou min-heap.

**Merge Sort:**

Baseado no conceito de divisão e conquita, este algoritmo segrega o vetor em duas partes de forma contínua até separar totalmente os elementos, uma vez que um veto de apenas um elemento é considerado ordenado. Assim, logo em seguida cada elemento é comparado com o seu adjacente e esses elementos são ordenados para formarem pares de elementos. Esse processo é feito de forma contínua, comparando os pares formados anteriormente com os seus pares adjacentes e combinando-os até que a extinção de todos os subconjuntos do vetor principal.

**Funcionamento dos métodos:**

**ordenaA**: este método recursivo determina as posições de início e fim que indicam os locais para a divisão do vetor em duas metades de forma contínua até que o parâmetro início seja maior ou igual que o parâmetro fim. Após duas chamadas de si mesmo, este método passa as variáveis indicativas das posições do vetor para o **ordenaB.**

**ordenaB**: método para realizar as comparações dos items que estão nas posições indicadas (início, meio, meio+1 e fim) para verificar qual item é maior que outro. Feita essas comparações, os itens ordenados são combinados e alocados em um vetor auxiliar para que ao fim sejam organizados no vetor original.

Dessa forma, considerando o funcionamento dos dois métodos descritos acima, o algoritmo de ordenação corresponde ao **Merge Sort**.