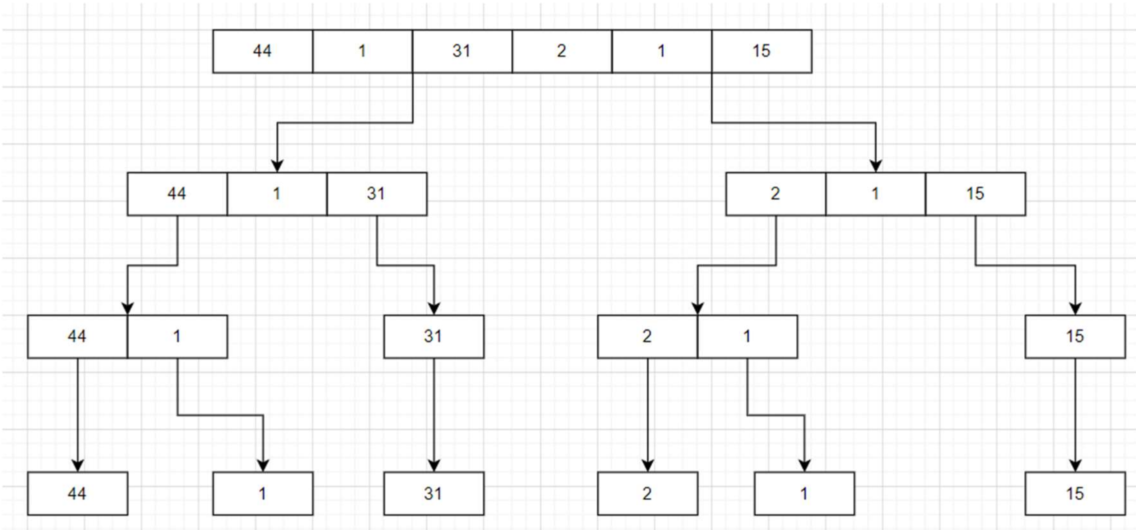
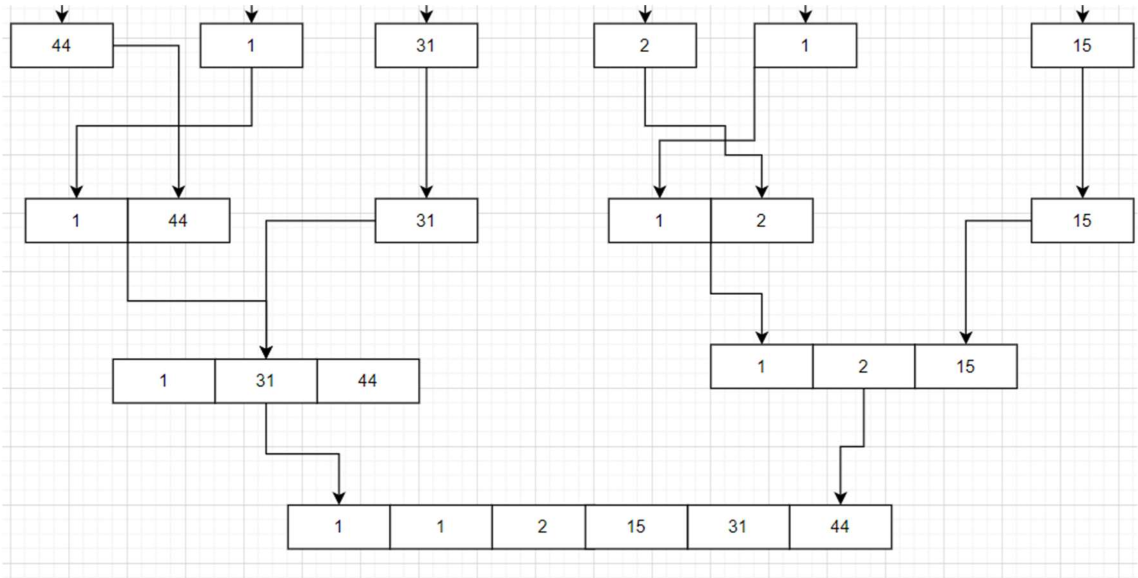
Entendendo o comportamento:



A primeira fase do algoritmo irá dividir a sequência de números em metades, até que todos estejam divididos em grupos contendo apenas 1 número, esta sendo a parte “dividir” do “dividir e conquistar”



Após a divisão, os números são novamente juntados e ordenados, visto que os subgrupos sendo unidos já estão ordenados, é apenas comparada a ponta e cada um dos grupos, para verificar qual deve ser escrito primeiro.

Analisando o comportamento do “dividir e conquistar” percebemos que ele divide um problema de tamanho N em 2 problemas de tamanho N/2

Portanto teremos $\mathrm{O}\frac{N}{2} + \mathrm{O}\frac{N}{2} \; = \; \mathrm{O}(2\frac{N}{2})$

E esta equação pode ser dada a sua equivalente: T(N) = $2\mathrm{T}\frac{N}{2}$

E para a parte de reunificação do problema temos: O(n) = n

Juntando ambas as partes, temos a complexidade total, que fica $T(N) = 2T\frac{N}{2} + O(N)$

Como citado previamente, o problema será dividido em dois subproblemas até ser alcançado grupos de 1 único número, portanto teremos:

$$\frac{N}{\frac{2}{2}} = \frac{N}{4}$$

$$T(N) = 2T\frac{N}{2} + O(N)$$

$$T(N/2) = 2T\frac{N}{4} + N/2 + N$$

$$T(N/2) = 4T\frac{N}{4} + 2N$$

$$T(N/4) = 2T\frac{N}{8} + N/4$$

.

.

.

Para resolver essa recorrência, adaptamos para:

N = 2^k, e K= log n

Assim temos:

T(n) = 2^k * T(n/2^k)+ k*n

Portanto:

T(n) = n*T(1)+n log n

= O(n log n)

Complexidade do melhor e pior caso:

Em qualquer sequência de números, ordenados ou não, o algoritmo irá realizar o mesmo número de passos, sendo: dividir o problema até o caso base, comparar os subgrupos e os unir novamente, portanto o pior e melhor casos tem a mesma complexidade de tempo: O(n log n), visto que o melhor e pior casos tem a mesma complexidade, esta também é a complexidade do caso médio :)