Iterative Approach

Ex 1

$T(n) = T(n-1) + 1 \longrightarrow T(n) = \begin{cases} 1, & n = 0 \\ T(n-1)+1, & n > 0 \end{cases}$

$$\begin{aligned} T(n) &= (T(n-2)+1)+1 \\ &= T(n-2) + 2 \\ &= T(n-3) + 1 + 2 \\ &= T(n-3) + 3 \\ &\vdots \\ &= T(n-n) + n \\ &= T(0) + n \\ &= 1 + n \end{aligned}$$

$\longrightarrow$ will stop when $T(0) \Rightarrow T(n-n)$

$n = 0 \Longrightarrow T(n) = 1$

$\Rightarrow T(n) = O(n)$

Merge Sort

$$\begin{aligned} T(n) &= 2T(n/2) + n \\ &= 2(2T(n/4) + n/2) + n \\ &= 4T(n/4) + n + n \\ &= 4T(n/4) + 2n \\ &= 4(2T(n/8) + n/4) + 2n \end{aligned}$$

$2n \longrightarrow (\log 4)\, n$

$$\begin{aligned} &= 8T(n/8) + n + 2n \\ &= 8T(n/8) + 3n \end{aligned}$$

$3n \longrightarrow \log(8)\, n$

$$\vdots$$

$$= 2^k T(n/2^k) + kn$$

will go on till $\frac{n}{2^k} = 1 \Rightarrow n = 2^k$

$\Rightarrow k = \log n$

$$\Rightarrow 2^{\log_2 n}\, T(1) + (\log n)\, n$$

$$\Rightarrow n \cdot T(1) + n \log n$$

$$\Rightarrow n + n \log n$$

$$\Rightarrow T(n) = O(n \log n)$$

(2) Recursive Tree Method

$$T(n) = 2T(n/2) + n$$

Recursive call	# of nodes	Tree	Total operation (for merging)
$T(n)$ $i=0$	$1 = 2^0$	n	n
$T(n/2)$ $i=1$	$2 = 2^1$	$n/2 \quad n/2$	$n/2 + n/2 = n$
$T(n/4)$ $i=2$	$4 = 2^2$	$n/4 \quad n/4 \quad n/4 \quad n/4$	$n/4 + n/4 + n/4 + n/4 = n$
⋮	⋮		
$T(n/2^i)$	2^i	. . . . . .	$2^i \times n/2^i = n$

levels $= \log_2 n$

on stops when $2^i = 1$

$i = \log_2 n$

Total operations

$$\sum_{i=0}^{\log_2 n} n$$

$$= n \sum_{i=0}^{\log_2 n}$$

$$= n(1 + 1 + \cdots + 1)$$

$$= n \log n$$

$$T(n) = O(n \log n) \checkmark$$

Quick Sort

worst case $\longrightarrow$ pivot always max/min
↳ one partition empty
↳ partition worst case $\longrightarrow O(n)$

$$T(n) = T(0) + T(n-1) + O(n)$$

$$= O(1) + T(n-1) + O(n)$$
$$= T(n-1) + n$$

$$= T(n-2) + 2n$$
$$= T(n-3) + 3n$$
$$\vdots$$
$$= T(n-k) + kn$$

will stop when $n-k=0 \longrightarrow k=n$

$$T(n) = T(0) + n \times n$$
$$= 1 + n^2$$

$$T(n) = O(n^2)$$