Master Theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n) \qquad a \geq 1 \quad b > 1$$

$$\boxed{\text{compare } f(n) \text{ to } n^{\log_b a}}$$

Cases:

(1) $f(n) \in O\left(n^{\log_b a - \varepsilon}\right) \quad \varepsilon > 0 \quad T(n) \in \Theta\left(n^{\log_b a}\right)$

(2) $f(n) \in \Theta\left(n^{\log_b a}\right) \qquad T(n) \in \Theta\left(n^{\log_b a} \cdot \log n\right) = \Theta\left(f(n) \cdot \log n\right)$

(3) $f(n) \in \Omega\left(n^{\log_b a + \varepsilon}\right) \quad \varepsilon > 0 \quad T(n) \in \Theta\left(f(n)\right)$

if the regularity condition holds for $f(n)$

$$\boxed{\text{regularity cond:} \quad a f\left(\frac{n}{b}\right) \leq c f(n) \quad c < 1}$$

2

$$T(n) = T\left(\frac{2n}{3}\right) + 1$$

$a = 1$

$b = \frac{3}{2}$

$f(n) = 1$

$n^{\log_{3/2} 1} = n^0 = 1$

$$f(n) = 1 \in \Theta(1) = \Theta\left(n^{\log_{3/2} 1}\right)$$

Case (2) of MT: $T(n) \in \Theta(\log n)$

$$T(n) = T\left(\frac{n}{2}\right) + 1$$

Binary Search

③ $T(n) = 3T\left(\frac{n}{4}\right) + n \cdot \log n$

$a = 3$
$b = 4$

$\boxed{f(n) = n \log n}$

$n^{\log_4 3}$

$$\log_4 2 < \underbrace{\log_4 3}_{} < \log_4 4$$
$$\frac{1}{2} < \qquad < 1$$

$\log_4 3 + \varepsilon < 1$

$0 < \varepsilon < 1 - \log_4 3$

$$\sqrt{n} < n^{\log_4 3} < n \ll n \log n = f(n)$$

$f(n) = \Omega\left(n^{\log_4 3 + \varepsilon}\right)$

(3) case if regularity holds

$$a f\left(\frac{n}{b}\right) \leq c f(n) \qquad f(n) = n \log n$$

$a = 3$
$b = 4$

$$\log \frac{x}{y} = \log x - \log y$$

$$\underbrace{3\left(\frac{n}{4}\right) \log\left(\frac{n}{4}\right)}_{\frac{3}{4} \cdot n \left(\log n - \log 4\right)} \leq c \cdot n \log n$$

$$\frac{3}{4} \cdot n \left(\log n - \log 4\right) < \frac{3}{4} \cdot n \log n \leq c\, n \log n$$

$c = \frac{3}{4}$ is a good choice

$c < 1$

The reg. cond. holds

$\Longrightarrow$ Case (3) of M.T.

$$T(n) \in \Theta(\underbrace{n \log n}_{f(n)})$$

4. $T(n) = 2T\left(\frac{n}{2}\right) + n\log n$

$a = 2$
$b = 2$

$f(n) = n\log n$

$n^{\log_2 2} = n$

$f(n) = n\log n \in \Omega(n)$

for (3) we need: $\underbrace{f(n) = n\log n \in \Omega(n^{1+\varepsilon})}_{\text{This is not true for any } \varepsilon > 0.}$ for some $\varepsilon > 0$

Assume on the contrary that there is such an $\varepsilon > 0$ that

$$n \log n \in \Omega(n^{1+\varepsilon})$$

So $\exists\, \boxed{c > 0}$ $n \log n \geq c \cdot n^{1+\varepsilon}$ for $n \geq n_0$

$$\boxed{\frac{\log n}{n^{\varepsilon}} \geq c}$$

$\log n \xrightarrow{n \to \infty} \infty$ $\left(\lim\limits_{n \to \infty} \log n = \infty \right)$

$n^{\varepsilon} \longrightarrow \infty$ $\varepsilon > 0$

$\lightning$

L'Hospital

$g(x), h(x) \xrightarrow[x \to \infty]{} \infty$

$$\lim_{x \to \infty} \frac{g(x)}{h(x)} = \lim_{x \to \infty} \frac{g'(x)}{h'(x)}$$

$$\boxed{\lim \frac{\log n}{n^{\varepsilon}}} = \lim \frac{\frac{1}{n}}{\varepsilon \cdot n^{\varepsilon - 1}} = \lim_{n \to \infty} \frac{1}{\varepsilon \cdot n^{\varepsilon}} = 0$$

$f(n) = n \log(n) \notin \Omega(n^{1+\varepsilon})$ MT cannot be used.

For practice:

1. $T(n) = 2T\left(\frac{n}{2}\right) + \begin{cases} 1 \\ n \\ n^2 \\ n^3 \end{cases}$

2. $T(n) = T\left(\frac{9n}{10}\right) + n$

3. $T(n) = 16T\left(\frac{n}{4}\right) + n^2$

4. $T(n) = 7 \cdot T\left(\frac{n}{3}\right) + n^2$

5. $T(n) = 7 \cdot T\left(\frac{n}{2}\right) + n^2$

6. $T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n}$

Counting inversions in an array $A[1:n]$
while performing MergeSort.

D&C:

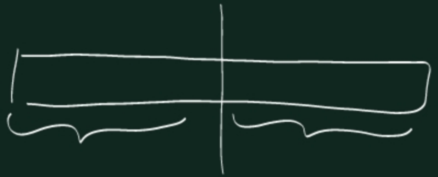
1. If the array is very small ($n=1$) solve in one step.

If not then divide the array into 2 subarrays.

$$m := \left\lfloor \frac{1+n}{2} \right\rfloor \longrightarrow A[1:m] \text{ and } A[m+1:n]$$

more general: $A[p:r]$ $\quad q := \left\lfloor \frac{p+r}{2} \right\rfloor \longrightarrow A[p:q]$ and $A[q+1:r]$

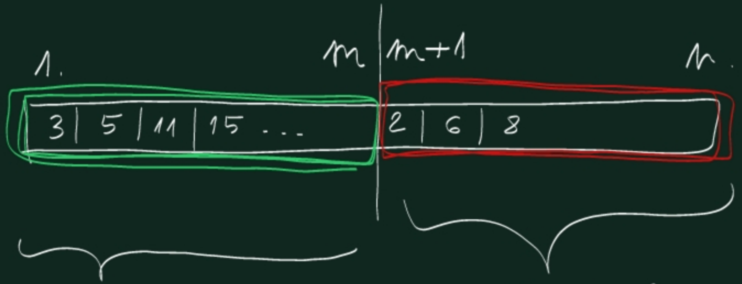
② Solve the problem on the 2 subarrays with recursive calls of the algorithm.



③ Combine the answers to give a final answer.

Our algorithm <u>sorts</u> the array <u>and</u> <u>counts</u> the inversions.

$A[1:n]$



Sorted

# of inversions here are <u>inv1</u>

Sorted

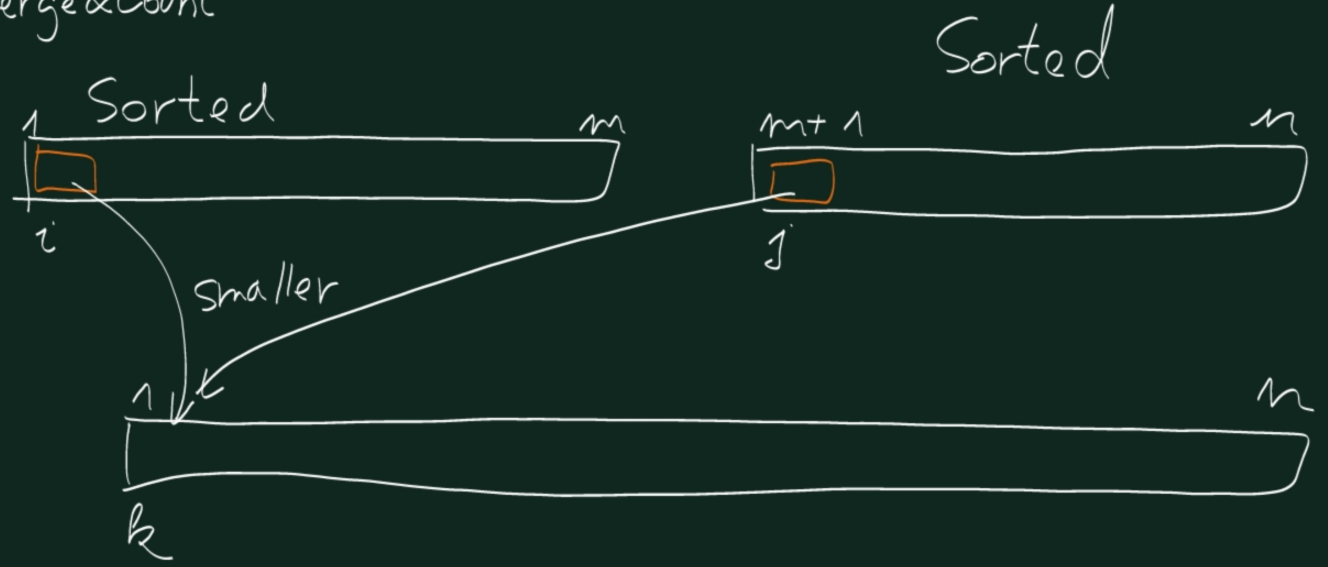
inversions here = <u>inv2</u>

The # of inversions in $A[1:n]$ is equal to

$$\text{inv1} + \text{inv2} + \#\text{ of } (\square, \square) \text{ inversions}$$

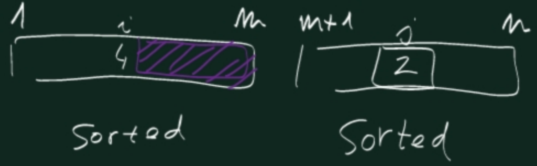
We count the # of ($\square$, $\square$) inversions along the Merge procedure.

Merge&Count



When merging the sorted subarrays $A[1:m]$ and $A[m+1:n]$

we use indeces $1 \le i \le m$ and $m+1 \le j \le n$.



1. If $A[i] < A[j]$ then there are no inversions here

2. If $A[i] > A[j]$ then $(A[i], A[j])$ are in inversion

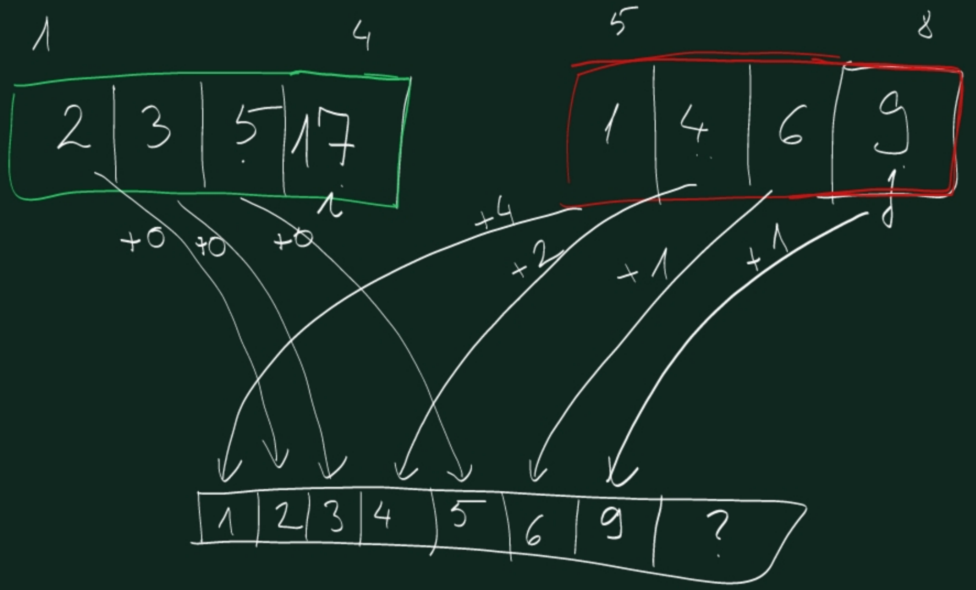
   <u>and</u> $A[j]$ is in inversion with

   $A[i+1], A[i+2], A[i+3], \dots, A[m]$ (all of)

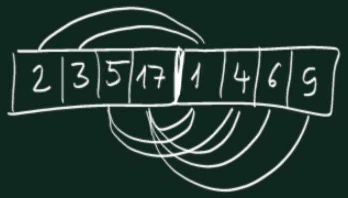
   [BECAUSE $A[i] \le A[i+1] \le A[i+2] \le \dots \le A[m]$]

$m-(i-1) = m-i+1$

We have found $m-(i-1)$ inversions.



8 inv:



$$\mathrm{inv}3 := 0$$

$$\left.\begin{array}{r} +4 \\ +0 \\ +0 \\ +2 \\ +0 \\ +1 \\ +1 \end{array}\right\} = 8$$

$$\mathrm{Merge\&Count}(A, \overbrace{p, r}^{\text{sorted}}, \overbrace{r+1, q}^{\text{sorted}}) : \mathbb{N}$$

number of ( ) inversions

$i := p$, $j := r+1$, $k := p$, $inv := 0$

copy A to B [for $i = p$ to q, $B[i] := A[i]$]

While $(i \le r) \wedge (j \le q)$

  if $B[i] \le B[j]$ then $A[k] := B[i]$ $i := i+1$ $k := k+1$

  else $A[k] := B[j]$ $\boxed{inv := inv + r - (i-1)}$ $j := j+1$ $k := k+1$

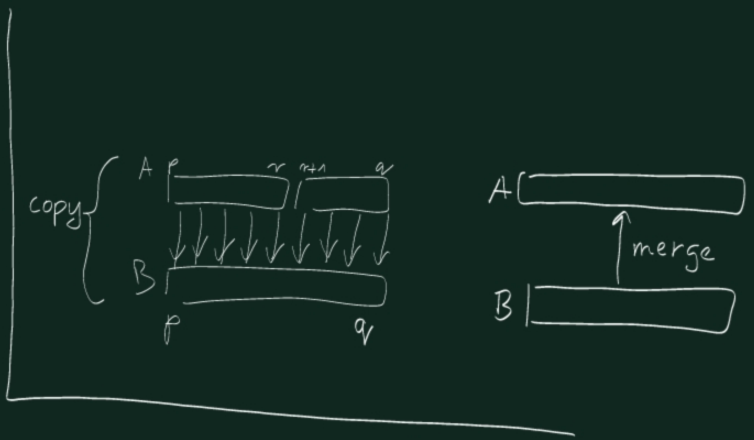
While $(i \le r)$

  $A[k] := B[i]$ $i := i+1$, $k := k+1$

While $(j \le q)$

  $A[k] := B[j]$ $j := j+1$; $k := k+1$

Return $\boxed{inv}$



$\text{CountInversions}(A, p, q) : \mathbb{N}$

if $p = q$ return 0

else $r := \left\lfloor \frac{p+q}{2} \right\rfloor$

$inv1 := \text{CountInversions}(A, p, r)$

$inv2 := \text{CountInversions}(A, r+1, q)$

$inv3 := \text{Merge\&Count}(A, p, r, r+1, q)$

return $(inv1 + inv2 + inv3)$