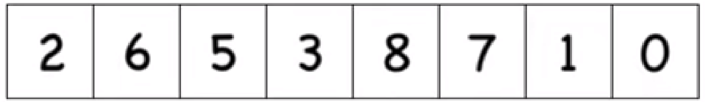
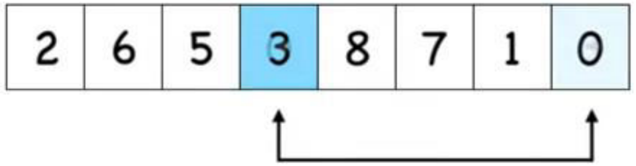
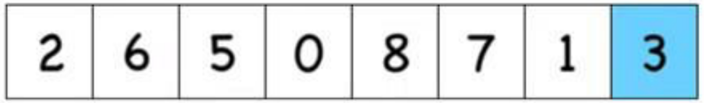
Quick Sort

1. Correct position in final, sorted array
2. Items to the left are smaller
3. Items to the right are larger

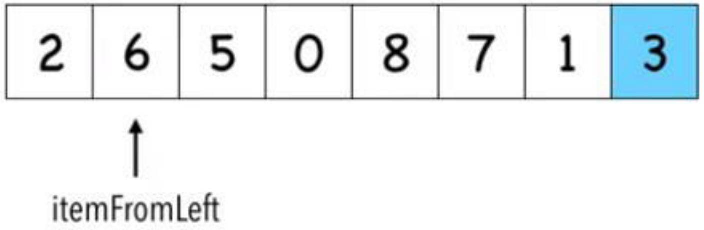




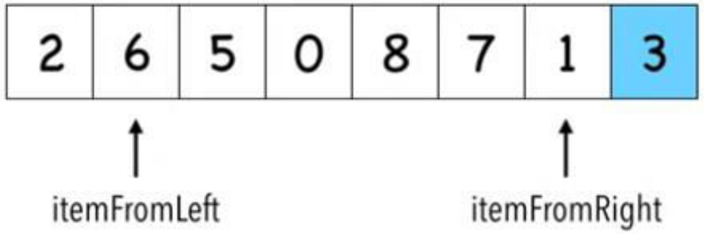
1. itemFromLeft that is larger than pivot

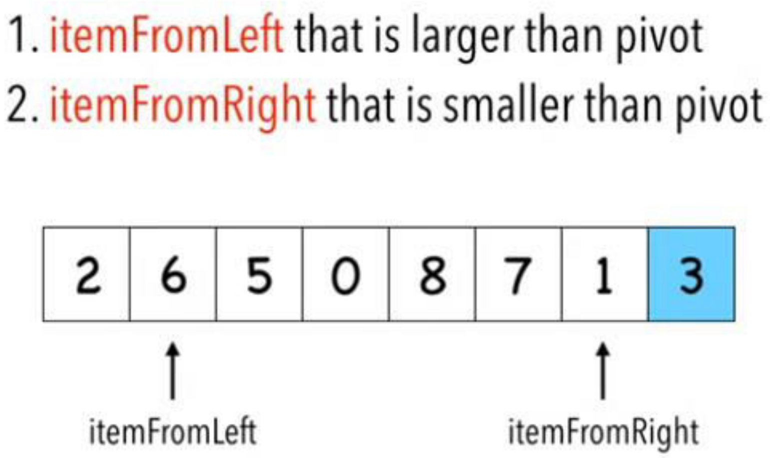


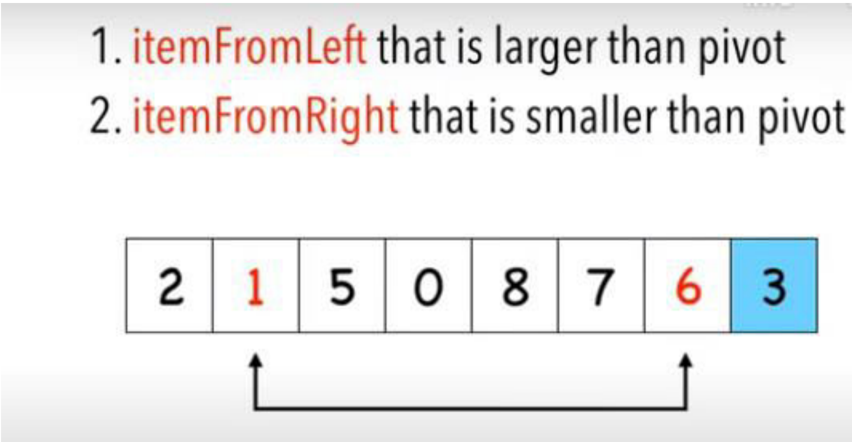
1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot

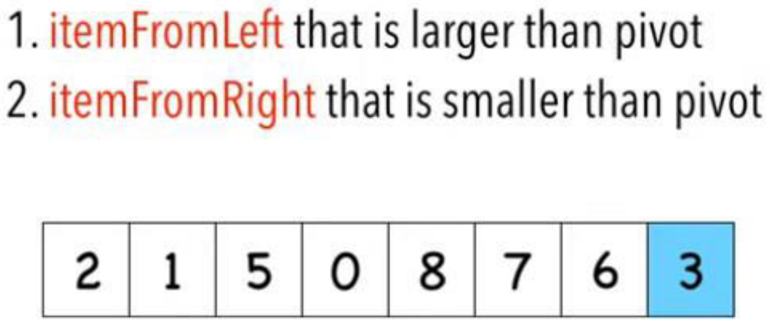


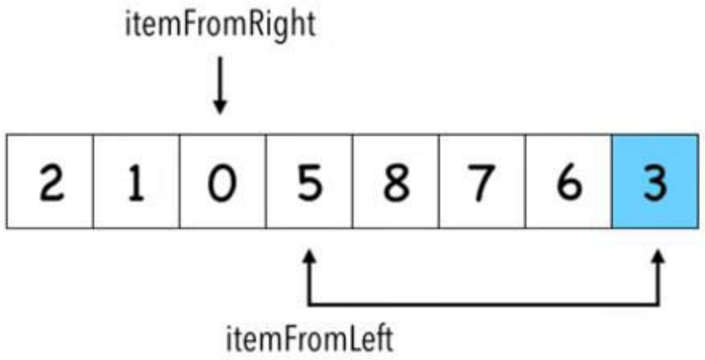
1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot

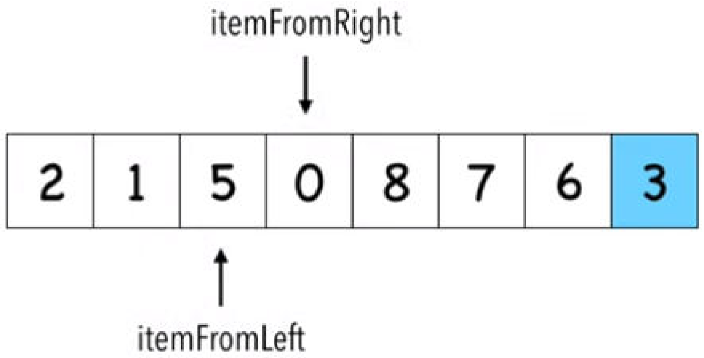



1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot
2 6 5 0 8 7 1 3
itemFromLeft
itemFromRight

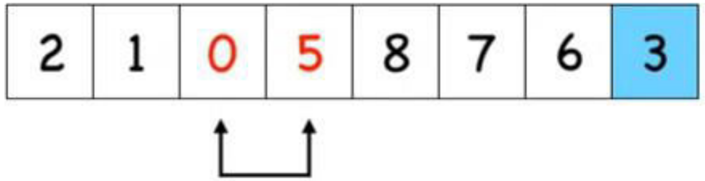

1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot
2 1 5 0 8 7 6 3

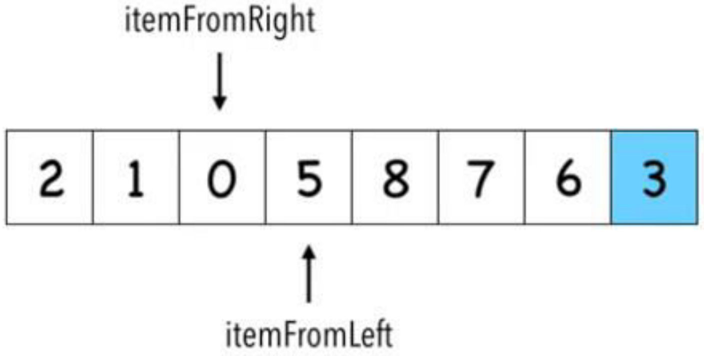

1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot
2 1 5 0 8 7 6 3





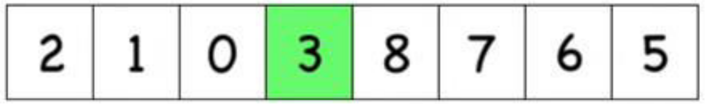
Swap itemFromLeft and pivot



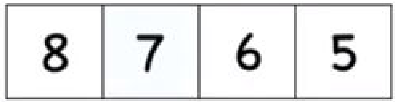


Stop when index of itemFromLeft > index of itemFromRight

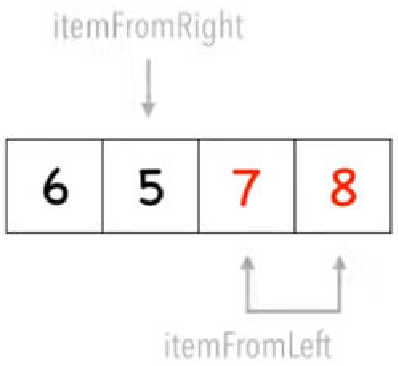
1. Correct position in final, sorted array
2. Items to the left are smaller
3. Items to the right are larger



How to choose pivot:
Median position element:



1. itemFromLeft that is larger than pivot
2. itemFromRight that is smaller than pivot



Worst Case Complexity: $O(n^2)$
Average Case Complexity: $\Theta(n \log n)$

Master's method is a quite useful method for solving recurrence equations because it directly gives us the cost of an algorithm with the help of the type of a recurrence equation and it is applied when the recurrence equation is in the form of:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

where, $a \geq 1$, $b > 1$ and $f(n) > 0$.

For example,
$c + T\left(\frac{n}{2}\right) \rightarrow a = 1$, $b = 2$ and $f(n) = c$,
$n + 2T\left(\frac{n}{2}\right) \rightarrow a = 2$, $b = 2$ and $f(n) = n$, etc.

T(n) = aT(n/b) + f(n), where, n = size of input a = number of subproblems in the recursion n/b = size of each subproblem. All subproblems are assumed to have the same size. Recurrence relation helps in finding the subsequent term (next term) dependent upon the preceding term (previous term).

Master's Theorem

Taking an equation of the form:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

where, $a \geq 1$, $b > 1$ and $f(n) > 0$

The Master's Theorem states:

- **CASE 1** - if $f(n) = O(n^{\log_b a - \epsilon})$ for some $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- **CASE 2** - if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$
- **CASE 3** - if $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

By the use of these three cases, we can easily get the solution of a recurrence equation of the form $T(n) = aT\left(\frac{n}{b}\right) + f(n)$.

Example 1

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

Here, $a = 2$, $b = 2$, $\log_b a = \log_2 2 = 1$
Now, $n^{\log_b a} = n^{\log_2 2} = n$
Also, $f(n) = n$
So, $n^{\log_b a} = n = f(n)$
(comparing $n^{\log_b a}$ with $f(n)$) $=> f(n) = \Theta(n^{\log_b a})$
So, case 2 can be applied and thus $T(n) = \Theta(n^{\lg_b a} \lg n) = \Theta(n \lg n)$.

Example 2

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

Here, $a = 2$, $b = 2$, $\log_2 2 = 1$
$=> n^{\lg_b a} = n^1 = n$
$Also, f(n) = n^2$
$=> f(n) = \Omega(n^{1+\epsilon})$ ($\epsilon = 1$) (comparing $n^{\log_b a}$ with $f(n)$)
Case 3 can be applied if rest of the conditions of case 3 gets satisfied for $f(n)$.

The condition is $af(n/b) \leq cf(n)$ for some $c < 1$ and all sufficiently large n.
For a sufficiently large n, we have,
$af\left(\frac{n}{b}\right) = 2f\left(\frac{n}{2}\right) = 2\frac{n^2}{4} = \frac{n^2}{2} \leq \frac{1}{2}(n^2)$ (for $c = \frac{1}{2}$)

So, the condition is satisfied for $c = \frac{1}{2}$. Thus, $T(n) = \Theta(f(n)) = \Theta(n^2)$

Example 3

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

Here, $a = 2$ $b = 2$ $\log_2 2 = 1$

$n^{\log_2 2} = n$

$f(n) = \sqrt{n}$

$f(n) = O(n^{1-\epsilon})$ (Case I)

$T(n) = \Theta(n)$

Example 4

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

Here, $a = 3\ b = 4\ \log_4 3 = 0.792$
$f(n) = \Omega(n^{\log_4 3+\epsilon})$ (Case 3)
$3\left(\frac{n}{4}\right)\lg\left(\frac{n}{4}\right) \leq \frac{3}{4}n\lg n = c * f(n)$, $c = \frac{3}{4}$
So, $T(n) = \Theta(n\lg n)$

Example 5

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

Here, $a = 2$ $b = 2$ $\log_2 2 = 1$
$n^{\log_2 2} = n^1$
$f(n) = n \lg n$
$f(n)$ must be polynomially larger by a factor of n^ϵ but it is only larger by a factor of $\lg n$.
So, Master's theorem can't be applied.