Master's theorem:

1. To find time complexity of the Recursive algorithms.

2. Divide and conquer algorithms.

aT(n/b) + f(n)

n -> size

a -> No.of subproblems

n/b -> size of each subproblem

f(n) -> timecomplexity of the non-recusive algorithm.

Recursive equation should be in form below

$$T(n) = aT(n/b) + \theta(n^k \log^p n)$$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 1: $a > b^k$ (or) $\log_b a > k$

1. $T(n) = 2T(n/2) + 1$

$a = 2$
$b = 2$
$k = 0$
$p = 0$

$\log_2 2 > 0$

$1 > 0$

$T(n) = O(n^{\log_b a})$

$= O(n^{\log_2 2})$

$= O(n^1)$

$T(n) = O(n)$

Case 1: $a > b^k$ (or) $\log_b a > k$

2. $T(n) = 4T(n/2) + n$

$a = 4$
$b = 2$
$k = 1$
$p = 0$

$\log_2 4 > 1$

$2 > 0$

$T(n) = O(n^{\log_b a})$

$= O(n^{\log_2 4})$

$= O(n^2)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 1: $a > b^k$ (or) $\log_b a > k$

3. T(n) = 8T(n/2) + n

a = 8
b = 2
k = 1
p = 0

$\log_2 8 > 1$

$3 > 0$

$T(n) = O(n^{\log_b a})$

$= O(n^{\log_2 8})$

$= O(n^3)$

$T(n) = O(n^3)$

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

$\log_2 8 \Rightarrow \log_2 2^3 \Rightarrow 3 \log_2 2 \Rightarrow 3\,(1) = 3$

Case 1: $a > b^k$ (or) $\log_b a > k$

4. $T(n) = 9T(n/3) + n$

$a = 9$
$b = 3$
$k = 1$
$p = 0$

$\log_3 9 > 1$

$2 > 0$

$T(n) = O(n^{\log_b a})$

$= O(n^{\log_3 9})$

$= O(n^2)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \, \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \, \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

$\log_2 8 \Rightarrow \log_2 2^3 \Rightarrow 3 \log_2 2 \Rightarrow 3\,(1) = 3$

Case 1: $a > b^k$ (or) $\log_b a > k$

4. $T(n) = 7T(n/3) + n$

$a = 7$
$b = 3$
$k = 1$
$p = 0$

$\log_3 7 > 1$

$2 > 0$

$T(n) = O(n^{\log_b a})$

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 3: $a < b^k$ (or) $\log_b a < k$

1. $T(n) = 2T(n/2) + n^2$

a = 2
b = 2
k = 2
p = 0

2 < 2 ^ 2

2 < 4

T(n) = O(n^k log^p n)
= O(n^2 log^0n)
= O(n^2 * 1)
= O(n^2)

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \, log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \, loglogn)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 3: $a < b^k$ (or) $\log_b a < k$

2. T(n) = 2T(n/2) + n^2 / log^2 n

a = 2
b = 2
k = 2
p = -2

2 < 2 ^ 2

2 < 4

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \, \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \, \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 3: $a < b^k$ (or) $\log_b a < k$

3. T(n) = 2T(n/2) + n^2 log n

a = 2
b = 2
k = 2
p = 1

2 < 2 ^ 2

2 < 4

T(n) = O(n^k log^p n)
= O(n^2 log^1n)
= O(n^2 log n)

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 2: $a = b^k$ (or) $\log_b a = k$

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

a = 2
b = 2
k = 1
p = 0

2 = 2^1

2 = 2

$T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^{\log_2 2} \log^{0+1} n)$

$= O(n^1 \log^1 n)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \; log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \; loglogn)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 2: $a = b^k$ (or) $\log_b a = k$

2. T(n) = 2T(n/2) + n log n

a = 2
b = 2
k = 1
p = 1

2 = 2^1

2 = 2

$T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^{\log_2 2} \log^{1+1} n)$

$= O(n^1 \log^2 n)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 2: $a = b^k$ (or) $\log_b a = k$

3. $T(n) = 2T(n/2) + n / \log n$

$a = 2$
$b = 2$
$k = 1$
$p = -1$

$2 = 2^1$

$2 = 2$

$T(n) = O(n^{\log_b a} \log \log n)$
$= O(n^{\log_2 2} \log \log n)$
$= O(n^1 \log \log n)$
$T(n) = O(n \log \log n)$

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n\log_b a \; log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n\log_b a \; loglogn)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 2: $a = b^k$ (or) $\log_b a = k$

3. T(n) = 4T(n/2) + n^2 / log n

a = 2
b = 2
k = 1
p = -1

4 = 2^2

4 = 4

$T(n) = O(n^{\log_b a} \log \log n)$

$= O(n^{\log_2 4} \log \log n)$

$= O(n^2 \log \log n)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n \log_b a)$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n \log_b a \, \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n \log_b a \, \log \log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Case 2: $a = b^k$ (or) $\log_b a = k$

3. T(n) = 4T(n/2) + n^2 / log ^2 n

a = 4
b = 2
k = 2
p = -2

4 = 2^2

4 = 4

$T(n) = O(n^{\log_b a})$ (or) $O(n^k)$

$= O(n^{\log_2 4})$

$= O(n^2)$

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

Recursive equation should be in form below

$T(n) = aT(n/b) + \theta(n^k \log^p n)$

Where a≥1, b>1, k≥0 and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n\log_b a)$
2) If $a = b^k$ then
   a. If p > -1, then $T(n) = \theta(n\log_b a \; log^{p+1} n)$
   b. If p = -1, then $T(n) = \theta(n\log_b a \; loglogn)$
   c. If p < -1, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If p ≥ 0, then $T(n) = \theta(n^k log^p n)$
   b. If p < 0, then $T(n) = O(n^k)$

1. T(n) = 2T(n/2) + n log ^2 n
2. T(n) = 8T(n/2) + n^2
3. T(n) = 4T(n/2) + n^2 / log ^2 n
4. T(n) = 16T(n/2) + n ^2
5. T(n) = T(n/2) + n
6. T(n) = T(n/2) + 1
7. T(n) = 0.5T(n/2) + 1 -> cannot apply
8. T(n) = T(n) + 1 -> “

Recursive equation should be in form below

$$T(n) = aT(n/b) + \theta(n^k \log^p n)$$

Where $a \geq 1$, $b > 1$, $k \geq 0$ and p is a real no.

1) If $a > b^k$, then $T(n) = \theta(n^{\log_b a})$
2) If $a = b^k$ then
   a. If $p > -1$, then $T(n) = \theta(n^{\log_b a} \log^{p+1} n)$
   b. If $p = -1$, then $T(n) = \theta(n^{\log_b a} \log\log n)$
   c. If $p < -1$, then $T(n) = \theta(n^{\log_b a})$
3) If $a < b^k$ then
   a. If $p \geq 0$, then $T(n) = \theta(n^k \log^p n)$
   b. If $p < 0$, then $T(n) = O(n^k)$

Quick sort:

n	
n/2	n/2

$Q(n) = Q(n/2) + Q(n/2) + f(n)$

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

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

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

$A = 2$
$B = 2$
$K = 1$
$P = 0$

$a = b^k$
$2 = 2^1$

Case 2: sub case a.

$P > -1$ $T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^1 \log^1 n)$

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