Master Theorem

If $T(n) = aT([n/2] + O(n^d)$ for constant $a > 0$, $b > 1$ and $d >= 0$

$O(n^d)$	If $d > \log_b^a$
$O(n^d \log n)$	If $d = \log_b^a$
$O(n^{\log_b a})$	If $d < \log_b^a$

Application of master theorem:

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

Here, $a = 4$, $b = 2$, $d = 1$

$\mathrm{Log}_2^4 = 2$

Since $d < \log_b^a$, $T(n) = O(n^{\log_b a}) = O(n^3)$

2. $T(n) = 3T(n/2) + O(n)$ [karatsuba algorithm]

Here, $a = 3$, $b = 2$, $d = 1$

Log3 = 1.58 which is larger then d(1)

So $T(n) = O(n^{\log_a b})$

$= O(n^{1.58})$

3. $T(n) = 2T(n/2) + O(n)$

Here, $a = 2$, $b = 2$, $d = 1$

Log2 = 1 which is equal to d

So, $T(n) = O(n^1 \log n)$

$= O(n \log n)$

4. $T(n) = T(n/2) + O(1)$ [Binary Search]

Here, $a = 1$, $b = 2$, $d = 0$

Log1 = 0 which is equal to d

So $T(n) = O(n^0 \log n)$

$= O(\log n)$

5. $T(n) = 8T(n/2) + O(1)$

Here, $a = 1$, $b = 2$, $d = 0$

$\log 1 = 0$ which is equal to $d(1)$

So $T(n) = O(n^0 \log n)$

$= O(\log n)$