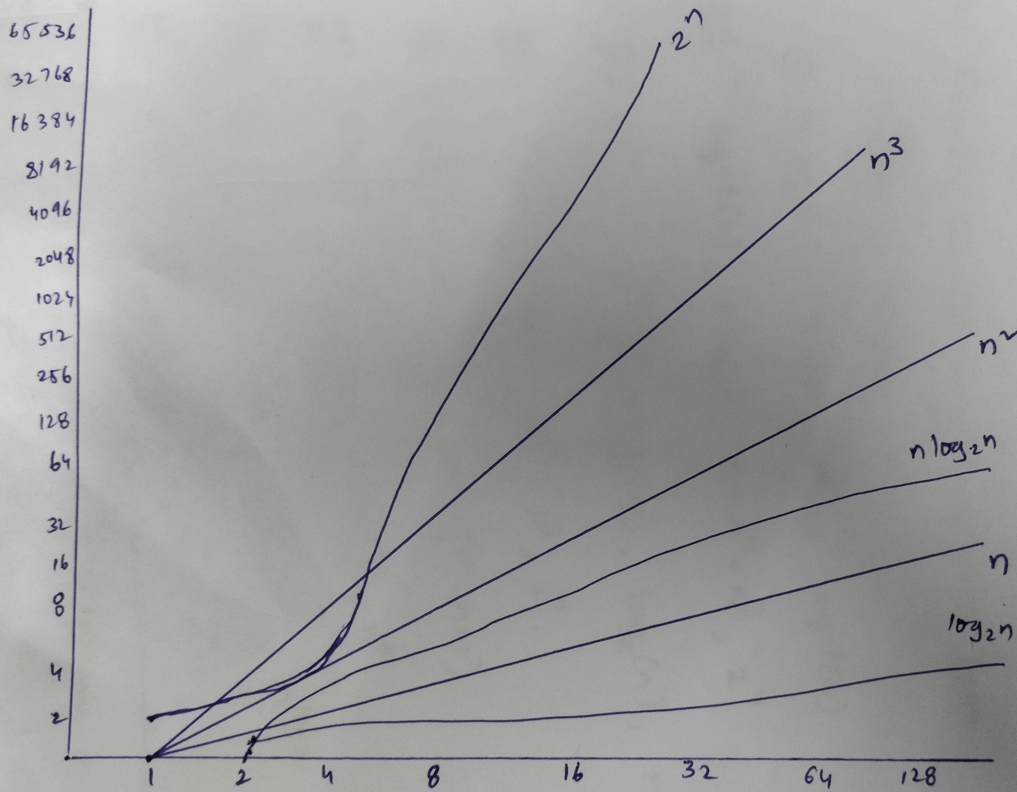
ANALYSIS OF ALGORITHMS

ORDER of Growth:-

Measuring the performance of an algorithm in relation with the input size n is called <u>order of growth</u>. For example, the order of growth for varying input size of n is as given below

n	logn	nlogn	n^2	2^n
1	0	0	1	2
2	1	2	4	4
4	2	8	16	16
8	3	24	64	256
16	4	64	256	65536
32	5	160	1024	4,294,967,296



Rate of growth of common computing time function

→ From the drawn graph, it is clear that the logarithmic function is the slowest growing function, and the exponential function 2^n is fastest and grows rapidly with varying input size n. The exponential function gives huge values even for small input n.

→ For instance; for the value of n = 16 we get $2^{16} = 65536$.

Q1. Arrange following rate of growth in increasing order

2^n, $n\log n$, n^2, 1, n, $\log n$, $n!$, n^3

Sol

$1, \log n, n, n\log n, n^2, n^3, 2^n, n!$

Q2. Reorder the following complexity from smallest to largest.

① $n\log_2(n)$, $n+n^2+n^3$, 2^4, sqrt(n)

Sol sqrt(n), $n\log_2 n$, $n+n^2+n^3$, 2^4

② n^2, 2^n, $n\log_2(n)$, $\log_2(n)$, n^3

Sol $\log n$, $n\log n$, n^2, n^3, 2^n.

③ $n\log(n)$, n^8, $n^2/\log n$, (n^2-n+1)

Sol $n\log n$, $\frac{n^2}{\log n}$, (n^2-n+1), n^8

④ $n!$, 2^n, $(n+1)!$, 2^{2n}, n^n, $n^{\log n}$.

Sol $n^{\log n}$, 2^n, $n!$, $(n+1)!$, 2^{2n}, n^n.

Q3. Arrange following functions n in increasing order:

2^n, $\log_2 n$, n^3, $n^{\log_2 n}$, $2^{\log_2 n}$, $n^2\log_2 n$, $e^{\log_2 n}$, 3^n, 2^{2^n}, $\frac{1}{n}$, $n\log_2 n$

Sol The increasing order will be

$\frac{1}{n} < \log_2 n < 2^{\log_2 n} < e^{\log_2 n} < n\log n < n^2\log_2 n < n^3 < n^{\log_2 n} < 2^n < 3^n < 2^{2^n}$