Order of Magnitude

=> To determine the time of Algo under Apriori Analysis.

=> Order of Magnitude of a statement of the Algo : refers to The count of fundamental operation in the step.

Eg.

```
Algo Sum (a,b,c)
{ integer a,b,c;
   1.  read (a,b)              ->   T.C = 1
   2.  if (a<b)                ->   T.C = 1
     {     c = a+b;     }      ->    ''    ''
     else
     {     c = a+b      }      ->    ''    ''

     Print (c)                 ->    ''    ''
```

T.C = constant

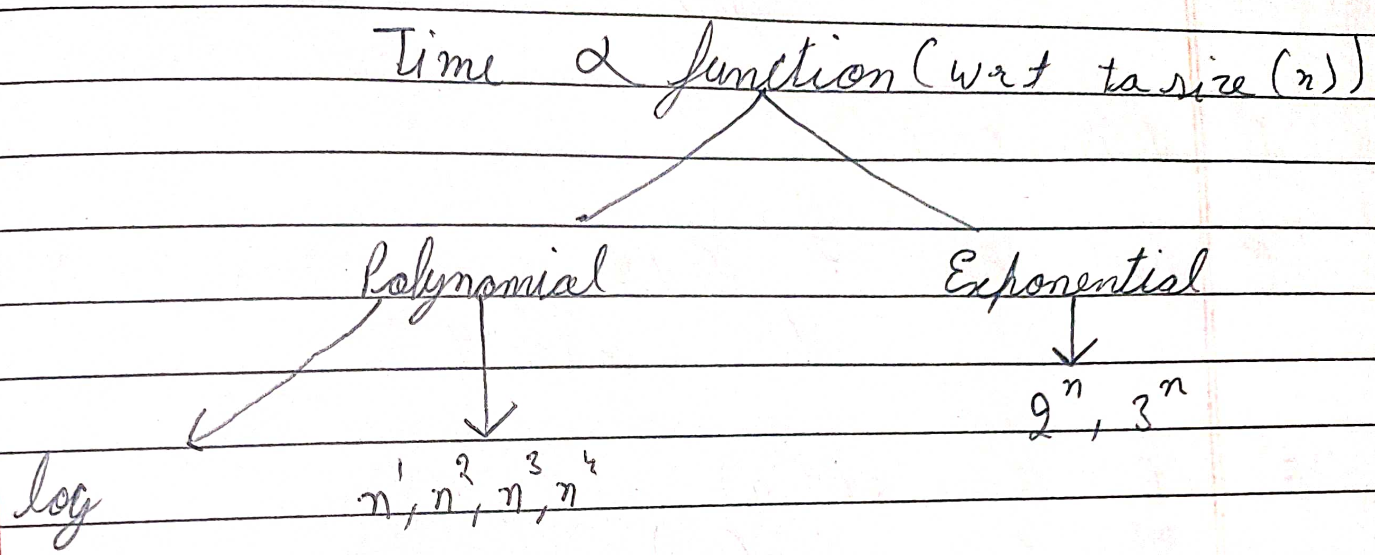
eg:

```
for 1 -> n
    c = a+b          ->  (+) = n times
                         (=) = n times
```

main (+)

so (n) = T.C

The basic objective of Apriori Analysis is to represent the T-C w.r.t input size (n)



=> Exponential fun. have higher rates of growth (Takes more time)

=> Polynomial fun. have lesser rate of growth (Take less time)

* T.C must be Polynomial to be Efficient. called Complexity Theory.

Apriori - Analysis.

① To determine the runing Time w.r.t increasing input size (n)

② To observe ~~for the~~ the Algo for diff. values of fixed (n).

Time - Complexity.

① Worst Case = ~~#~~ I/P depend in ~~less~~ max time of Algo.

② Avg. case = x

③ Best case => I/p for which Algo takes min time.

=> Best Case = B(n)
Worst Case = W(n)
Avg Case = A(n)
~~A~~ $B(n) \leq A(n) \leq W(n)$

① $B(n) = A(n) = W(n)$ => merge sort, Heap sort
② $B(n) < A(n) = W(n)$ => => linear sear, B.S
③ $B(n) = A(n) < W(n)$ => Quick sort.