int factorial of N:

int factorial (int n)
$$\frac{1}{2}$$

if $(n = 0)$ return $\frac{1}{2}$;

return factorial $(n-1) + n$;

 $\frac{1}{2}$
 $\frac{$

```
# Calculate Power fin:
 long power (inta, inta) { (am)
     if (n == 0) return 1;
    long halfpower = power (a, n/2);
    if (n%2 == 0) {
      return halfpower & halfpower;
   return halfpower * halfpower * a;
   } else {
T(N) = T(N/2) + 1
                        2nd
    = T (N/4)+2
                        ______38d
     = T(N/8) + 3
                         ---- 4th
    = T(N/16) + 4
                      ----- K+h
T(N) = T(N_{gk}) + K
       T(\frac{N}{2}x) = T(1) (base case \rightarrow n==0)
          N/2K = 1
             2^{K} = N
             k = \log_2(N)
```

$$T(N) = T(N/2)yN + \log N$$

$$= T(N) + \log N$$

$$= T(1) + \log N$$

$$T(N) = 1 + \log N$$

$$T$$

$$T(N) = 2^{K} T(N/2^{K}) + (2^{K}-1) \qquad K^{+}$$

$$T(N/2^{K}) = T(1)$$

$$\therefore N/2^{K} = 1$$

$$\therefore 2^{K} = N$$

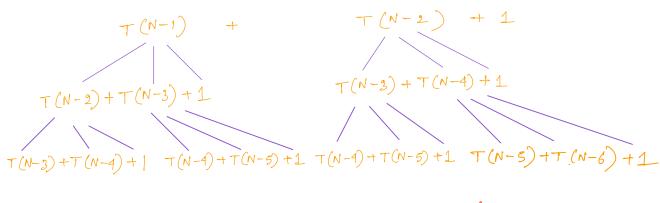
$$\therefore K = \log_{2}N$$

$$T(N) = 2^{\log_{2}N} T(N/2^{\log_{2}N}) + (2^{\log_{2}N}-1)$$

$$= N + T(1) + (N-1)$$

$$= N + N - 1$$

$$T(N) = 2N - 1$$



This seems to very complex if we want to calculate in Mathematical expression. So, we will try to calculate the worst performer function's time complexity to get the O(n).

A Here" T(n-1)" seems to be the slowest performer as it's value reduces very slowly.

A To calculate this we will assume to replace all for calls by the slowest for.

Substitution f^{n} max. TC term count O(1)1 T(n-1) $2 \rightarrow 2^{1}$ 1 T(n-2) $4 \rightarrow 2^{2}$ 3

 $T(N) = 2^{K} + (n-K) + 2^{K} - 1$