MODULE - 1

ADT - Algorithms & Properties - Proof of correctness - Algorithm analysis - Asymptotic notation & properties - Linear Search.

Max (a, n):

```
    max = a[0];
    for : i = 1 to i = n-1 do
    {
        if : a[i] > max , then
            m = a[i]
    }
    return (m);
```

Hypothesis: $n = K$, $K > n_0$
Proof : $n = K+1$ (T)

Correctness ⇒ PMI ⟶ Basic step, Inductive step.
⇒ Contradiction ⟶ Assume Conclusion wrong & Prove assumption is wrong.
⇒ Proof

Big O

$f : N \to +R$
$g : N \to +R$
if $f(n) = O(g(n))$

$f(n) \le c \cdot g(n)$, $n \ge n_0$, $\exists c \in R$
$n_0 \in N$

* $g(n)$ ⇒ UPPER BOUND
WORST CASE scenario.

$f(n)$ ⇒ ACTUAL TIME
$g(n)$ ⇒ ESTIMATED TIME

$f(n) = 4n+15$
Let $g(n) = n$

Let $C = 5$,
$f(n) \le c \cdot g(n)$ $\quad c = 5, n \ge n_0$
$4n+15 \le 5n$
$15 \le n$
$n_0 = 15$

$\therefore 4n+15 = O(n)$ $\quad c = 5,\ n \ge 15$

$f(n) = 4n^2 + 5n + 12$
Let $g(n) = n^2$

Let $C = 5$
$f(n) \le c \cdot g(n)$ $\quad c = 5, n \ge n_0$
$4n^2 + 5n + 12 \le 5n^2$
$5n \le -12$
$n \le -2$

$12 \le n^2 - 5n$
$12 \le n(n-5)$
$\therefore n = 12, 7$
$n_0 = 7$

$\therefore 4n^2 + 5n + 12 \le n^2$, $\forall n \ge 7$
$\therefore f(n) = O(n^2)$

* $O(f(n)) + O(g(n)) = O(\max\{f(n), g(n)\})$
* Any function is a order of itself
* Any Constant is order of 1

Big Ω

if $f(n) = \Omega(g(n))$; $f(n) \ge C \cdot g(n)$ $\forall n \ge n_0$
$g(n)$ = LOWER BOUND (BEST CASE Scenario)

Theta (Θ) :

$f(n) = \Theta(g(n))$, $\exists c_1, c_2 \in R$, $n_0 \in N$
$c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n)$

* It is a equivalence relation.
→ Reflexive
→ Symmetric
→ Transitive

TC is exact, we use Θ (AVG CASE Scenario)

$\lim_{n \to \infty} \frac{f(n)}{g(n)} \in R+$, then $f(n) \in \Theta(g(n))$

$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$, then $f(n) \in O(g(n))$
$g(n) \notin O(f(n))$

$\lim_{n \to \infty} \frac{f(n)}{g(n)} = +\infty$, then $f(n) \in \Omega(g(n))$
$g(n) \notin \Omega(f(n))$

$B(n) = \min\{T(I) \mid I \in D_n\}$
$W(n) = \max\{T(I) \mid I \in D_n\}$
$A(n) = \sum_I P(I) T(I)$

I - inputs
D_n - domain of n
T - TC of I
P - Avg prob.

#ABDUL BARI @ RECURSION :

① Void Test (int n) — T(n)

```
{
    if(n>0)
    {
        print("%d", n);   — 1
        Test(n-1);        — T(n-1)
    }
}
```

Tracing tree

$T(n) = \begin{cases} 1/\text{CONSTANT } c & , n = 0 \\ T(n-1) + 1 & , n > 0 \end{cases}$

① SUBSTITUTION METHOD
② RECURSIVE TREE
③ MASTERS THEOREM

I) Substitution method :

$T(n) = T(n-1) + 1$
$T(n) = [T(n-2) + 1] + 1$
$T(n) = T(n-2) + 2$

Sub n-1 in T(n),
$T(n-1) = T(n-2) + 1$
$T(n-2) = T(n-3) + 1$

$T(n) = [T(n-3)+1]+2$
$T(n) = T(n-3)+3$
⋮ × K times
$T(n) = T(n-K)+K$

$T(n) = T(n-K)+K$
Assume, $n-K=0$
$n=K$ @ $K=n$

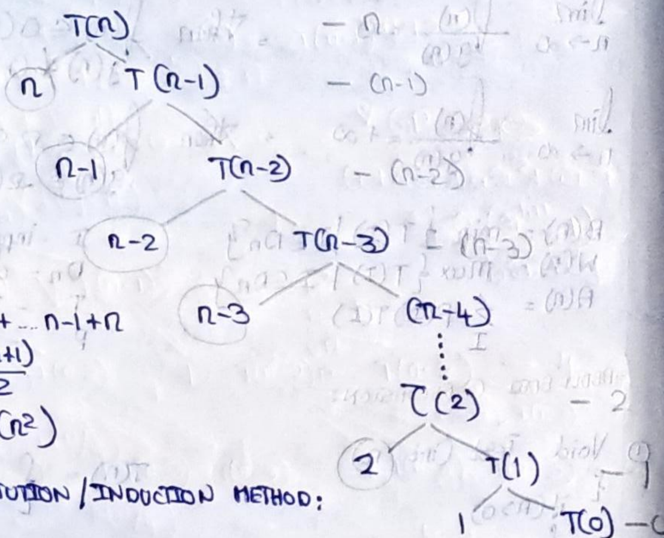
$T(n) = T(n-n)+n$
$T(n) = T(0)+n$
$T(n) = 1+n$

② Void test (int n) — T(n)
```
{ if (n>0)          — 
  { for (i=0; i<n; i++)   — n+1
        printf("%d", n);  — n
    Test(n-1);            — T(n-1)
  }
}
```

$T(n) = T(n-1)+2n+2$
$T(n) = T(n-1)+n$ ASN

$$T(n) = \begin{cases} \text{CONSTANT}=1 & n=0 \\ T(n-1)+n & n>0 \end{cases}$$

II) RECURSIVE TREE METHOD:



$\therefore T(n) = 1+2+\dots n-1+n$
$= \frac{n(n+1)}{2}$

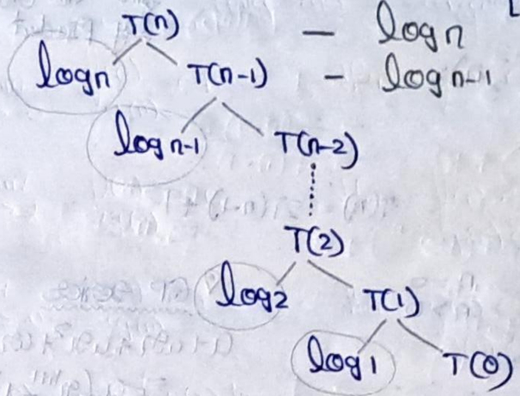
$T(n) = O(n^2)$

I) BACK SUBSTITUTION / INDUCTION METHOD:

$T(n) = T(n-1)+n$
$T(n) = T(n-2)+(n-1)+n$ $T(n-1) = T(n-2)+n-1$
$T(n) = T(n-3)+(n-2)+(n-1)+n$ $T(n-2) = T(n-3)+(n-2)$
⋮ K time
$T(n) = T(n-K)+(n-(K-1))+(n-(K-2))+\dots+(n-1)+n$
$= T(n-K)+.$
Assume $n-K=0$, $n=K$ $T(n) = T(0)+(n-n+1)+(n-n+2)+(n-1)+n$
$= T(0)+1+2+3+\dots(n-1)+n = 1+n(n+1)/2$

③ Void Test (int n) — T(n)
```
{ if (n>0)
  { for (i=1; i<n; i=i*2)   — log(n)
        printf("%d", i);
    Test(n-1);              — T(n-1)
  }
}
```
1
2
4
8
16

$T(n) = T(n-1)+\log(n)$ $\Rightarrow T(n) = \begin{cases} 1, & n=0 \\ T(n-1)+\log n, & n>0 \end{cases}$



$T(n) = \log n + \log(n-1) + \dots \log 2 + \log 1$
$= \log(n \times (n-1) \times \dots 2 \times 1)$
$= \log(n!)$
$= \log n!$

No tight bound for $\log n!$

$n! \to$ UB $\to n^n$
$n! \to$ LB

$\log n! \to$ UB $\to \log n^n$

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

$T(n) = T(n-1)+\log n$
$= T(n-2)+\log(n-1)+\log n$
$= T(n-3)+\log(n-2)+\log(n-1)+\log(n)$
⋮ K times
$T(n) = T(n-K)+\log(n-(K-1))+\log(n-(K-2))+\dots+\log(n-2)+\log(n-1)+\log n$
@ $n=K \Rightarrow T(n) = T(0)+\log 1+\log 2+\dots\log n$
$= \log n! = n\log n$

WKt $T(n) = T(n-1) + 1$ —— $O(n)$
$T(n) = T(n-1) + n$ —— $O(n^2)$
$T(n) = T(n-1) + \log$ —— $O(n \log n)$
$T(n) = T(n-1) + n^2$ —— $O(n^3)$
$T(n) = T(n-2) + 1$ —— $O(\frac{n}{2}) = O(n)$
$T(n) = T(n-100) + n$ —— $O(n^2)$

NOTE: !
* Here NOTE that $T(n-1)$ & other doesn't have COEFFICIENT.
* So we can easily find the order by product.

④ Algorithm test (int n) — $T(n)$

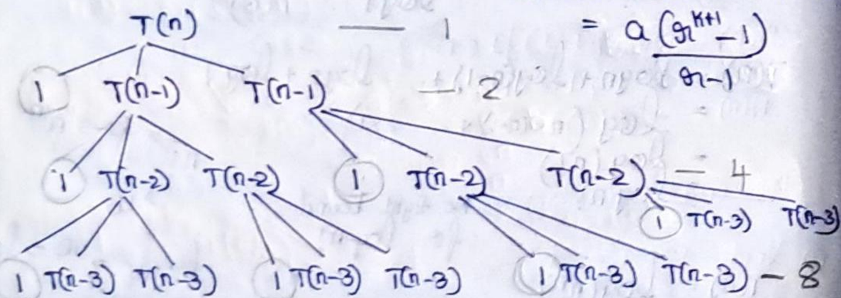
```
{
  if (n>0)
  {
    Printf("%d", n);      — 1
    test(n-1);            — T(n-1)
    test(n-1);            — T(n-1)
  }
}
```

$T(n) = 2T(n-1) + 1$

$$T(n) = \begin{cases} 1 & n = 0 \\ 2T(n-1) + 1 & n > 0 \end{cases}$$

GP Series

$a + ar + ar^2 + ar^3 + \ldots = \frac{a(r^{k+1} - 1)}{r-1}$



$T(0)$ — 2^k

r = ratio

$r = 2, a = 1$

$T(n) = 1 + 2^2 + 2^3 + 2^4 \ldots\ldots 2^k = \frac{1(2^{k+1} - 1)}{1} = 2^{k+1} - 1$

GP series

Assume $n - k = 0$
$n = k$ @ k

$\therefore T(n) = 2^{n+1} - 1$

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

$T(n) = 2T(n-1) + 1$ $\qquad 2^0 T(n - 2^0) + 1$
$T(n) = 2[2T(n-2) + 1] + 1$
$= 4T(n-2) + 2 + 1$
$= 4[2T(n-3) + 1] + 2 + 1$
$= 8T(n-3) + 4 + 2 + 1$
⋮ k times

$a = 1$
$r = 2$
$\Rightarrow GP = \frac{a(r^{k+1} - 1)}{r-1}$

$= 2^k T(n-k) + 2^{k-1} + \ldots + 2^2 + 2 + 1$
$= 2^k T(n-k) + \frac{1(2^{k+1} - 1)}{2-1}$
$= 2^k T(n-k) + 2^{k+1} - 1$

@ $n = k \Rightarrow T(n) = 2^n T(0) + 2^{n+1} - 1$
$= 2^{n+1} - 1$
$T(n) = O(2^n)$

Ⅲ MASTERS THEOREM:

ar^k

WKt $T(n) = 2T(n-1) + 1$ $\qquad T(n) = O(2^n)$
$T(n) = 3T(n-1) + 1$ $\qquad T(n) = O(3^n)$
$T(n) = 4T(n-1) + \log n$ $\qquad T(n) = O(4^n . \log n)$
$T(n) = 2T(n-1) + n$ $\qquad T(n) = O(2^n . n)$

General form of recurrence relation,

! $T(n) = aT(n-b) + f(n)$
$a > 0, b > 0$ and $f(n) = O(n^k)$ where $k \geq 0$

$T(n) = 2T(n-3) + 1$ $\qquad T(n) = O(2^{n/3})$

CASE:

1. if $a > 1 \Rightarrow T(n) = O(n^k . a^{n/b}) = O(f(n) a^{n/b})$
2. if $a < 1 \Rightarrow T(n) = O(n^k)$
$= O(f(n))$
3. if $a = 1 \Rightarrow T(n) = O(n^{k+1})$
$= O(n * f(n))$

RR Dividing functions

⑤ Algorithm test (int n) — T(n)
{
    if (n>1)
    {
        Printf("%d", n); — 1
        test (n/2); — T(n/2)
    }
}

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

$$T(n) = \begin{cases} 1 & n = 1 \\ T(n/2)+1 & n>1 \end{cases}$$

$T(n) = T(n/2) + 1$
$T(n) = T(n/4) + 2$
$T(n) = T(n/8) + 3$
$\vdots$ k×
$T(n) = T(n/2^k) + k$

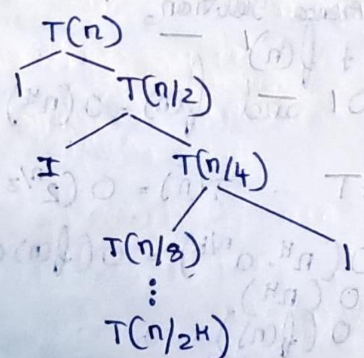
wkt $\frac{n}{2^k} = 1$
$n = 2^k$
$\log_2 n = k$

wkt
$n = k$ $(n-k=0)$
@ k ⇒ $T(n) = T(n2^{-n}) + n$

$\therefore T(n) = T(1) + \log_2 n$
$= 1 + \log_2 n$
$T(n) = O(\log_2 n)$

$a^b = c$
$b = \log_a c$

Reduce the Value of n ⇒ n, $n-b$, $\sqrt{n}$, $\frac{n}{2}$



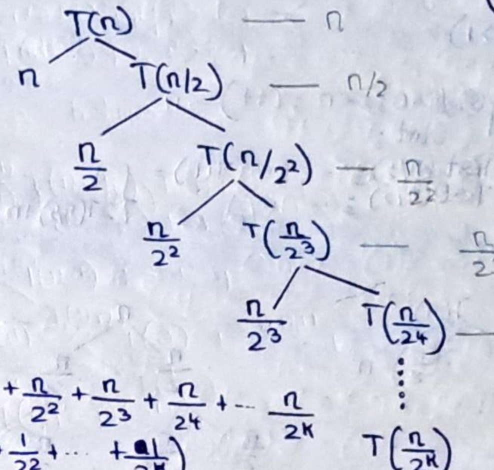
k steps
$\underbrace{1+1+\ldots+1}_{k \text{ terms}}$

$\frac{n}{2^k} = 1$
$n = 2^k$
$k = \log_2 n$

$T(n) = k$
$= O(\log_2 n)$

⑥ Let $T(n) = \begin{cases} 1 & n = 1 \\ T(n/2)+n & n>1 \end{cases}$

$\sum = 1$ circle.



$T(n) = n + \frac{n}{2} + \frac{n}{2^2} + \frac{n}{2^3} + \frac{n}{2^4} + \ldots \frac{n}{2^k}$

$= n\left(\frac{1}{2} + \frac{1}{2^2} + \ldots + \frac{1}{2^k}\right)$

$= \cancel{n(1-2^k}$ $= n \sum_{i=0}^{k} \frac{1}{2^i} = 1$

$= n \times 1$
$T(n) = O(n)$

$GP = \frac{a(r^{k+1}-1)}{r-1}$ $r>1$
$= \frac{a(1-r^{k+1})}{1-r}$

$T(n) = T(n/2) + n$
$= T\left(\frac{n}{2^2}\right) + \frac{n}{2} + n$
$= T\left(\frac{n}{2^3}\right) + \frac{n}{2^2} + \frac{n}{2} + n$
$= T\left(\frac{n}{2^4}\right) + \frac{n}{2^3} + \frac{n}{2^2} + \frac{n}{2} + n$
$\vdots$
$= T\left(\frac{n}{2^k}\right) + \frac{n}{2^{k-1}} + \frac{n}{2^{k-2}} + \ldots + \frac{n}{2} + n$
$= T\left(\frac{n}{2^k}\right) + n\left(\frac{1}{2^{k-1}} + \frac{1}{2^{k-2}} + \ldots + \frac{1}{2} + 1\right)$

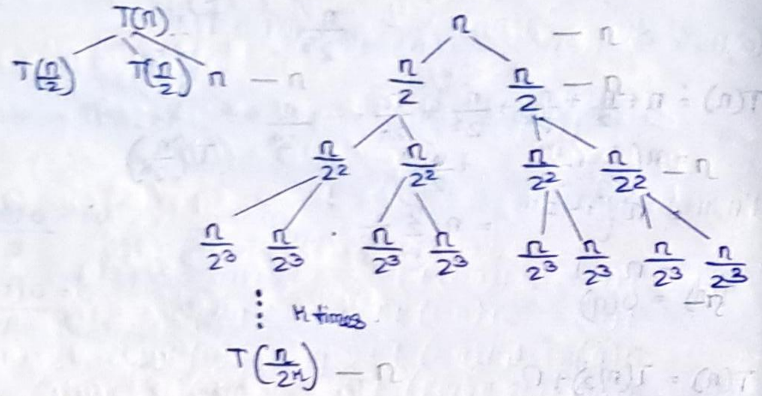
⇒

Assume $\frac{n}{2^k} = 1$ ⇒ $k = \log_2 n$ $\therefore T(n) = T(1) + n[1+1]$
$= 1 + 2n$
$T(n) = O(n)$

① $T(n) = 2T(n/2) + n$ — EASY { In dividing $f(n) \Rightarrow n=1$; decreasing $f(n) \Rightarrow n=0$

```
void test (int n)                    — T(n)
{
    if (n>1)
    {
        for (i=0; i<n; i++)          — n
            Stmt;
        Test (n/2);                  — T(n/2)
        Test (n/2);                  — T(n/2)
    }
}
```

$$T(n) = \begin{cases} 1 & , n=1 \\ 2T(n/2)+n & , n>1 \end{cases}$$



wkt $\frac{n}{2^k} = 1$

$k = \log_2 n$

$T(n) = nk$
$= n\log_2 n$

$T(n) = 2T\left(\frac{n}{2}\right) + n$

$= 2\left(2T\left(\frac{n}{2^2}\right) + \frac{n}{2}\right) + n$

$= 4T\left(\frac{n}{2^2}\right) + \frac{2n}{2} + n$

$= 4\left(2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}\right) + n + n$

$= 8T\left(\frac{n}{2^3}\right) + n + n + n$

⋮

$T(n) = 2^k\, T\left(\frac{n}{2^k}\right) + nk$

⇒ Assume, $\frac{n}{2^k} = 1$

$n = 2^k$ $k = \log_2 n$

∴ @ $n = k$

⇒ $T(n) = 2^k T(1) + nk$
$= 2^k + nk$
$= 2^k + nk$

$T(n) = n + n\log_2 n$
$= n(1+\log_2 n)$

$T(n) = O(n \cdot \log_2 n)$

! MASTER'S THEOREM FOR Dividing functions.

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$

$a \geq 1$, $b > 1$ $f(n) = O(n^k \cdot \log^p n)$ ① $\log_b a$ ② k

CASE-1: if $\log_b a > k$ then $\Theta(n^{\log_b a})$

CASE-2: if $\log_b a = k$

- if $p > -1$ then $\Theta(n^k \log^{p+1} n)$
- if $p = -1$ then $\Theta(n^k \log\log n)$
- if $p < 1$ then $\Theta(n^k)$

CASE-3: if $\log_b a < k$

- if ~~$\log_b a < k$~~ $p \geq 0$ then $\Theta(n^k \log^p n)$
- if $p < 0$ then $\Theta(n^k)$

CASE-I

⑧ $T(n) = 2\left(\frac{n}{2}\right) + 1$

$a = 2$
$b = 2$

$f(n) = \Theta(1) = (n^0 \cdot \log^0 n)$ ⇒ $k = 0$, $P = 0$

$\log_b a = 1$

Here $\log_b a > k$
⇒ $1 > 0$

CASE-1
$T(n) = O(n^{\log_2 2})$
$T(n) = O(n)$

⑨ $T(n) = 4T\left(\frac{n}{2}\right) + n$

$a = 4$
$b = 2$ $\log_b a = \log_2 4 = 2$

$f(n) = n = n^0 \log^0 n$ ⇒ $k = 0$, $P = 0$

Here $\log_b a > k$
⇒ $2 > 0$

CASE-1
$T(n) = O(n^{\log_a b})$
$= O(n2)$

⑨ $T(n) = 8T\left(\frac{n}{2}\right) + n$ Even if n^2 DOESN'T CHANGE

$a = 8$ $\log_2 8$ $K = 1$ $p = 0$
$b = 2$ $3 > 1$

CASE-1
$T(n) = O(n^3)$

CASE-2

⑩ $T(n) = 9T\left(\frac{n}{3}\right) + n^2$
$\log_3 9 = K = 2$ & $P = 0$
$\therefore T(n) = O(n^2 \log n)$ ← as it is

⑪ $T(n) = 2T\left(\frac{n}{2}\right) + n$
$\log_2 2 = K = 1$ & $P = 0$ EQUALS means
$\therefore T(n) = O(n^1 \log n)$ $f(n) \times \log n$
$P > -1$

⑫ $T(n) = 4T\left(\frac{n}{2}\right) + n^2$
$\log_2 4 = 2 = K$ & $P = 0$
$T(n) = n^2 \log n$

⑬ $f(n) = n^2 \log n$
$T(n) = n^2 \log n \times \log n$
$= n^2 \log^2 n$

⑭ $f(n) = n^2 \log^2 n$
$T(n) = n^2 \log^3 n$

⑮ $T(n) = 2T(n/2) + \frac{n}{\log n}$ as it is
@$P = -1$
$T(n) = n \cdot \log \log n$

⑯ $f(n) = \frac{n}{\log^2 n}$ when less than -1 take n as such
$T(n) = n$

CASE-III

⑰ $T(n) = T\left(\frac{n}{2}\right) + n^2$ @C3
$\log_b a = \log_2 1 = 0 < K = 2$
$T(n) = n^2$

⑱ $f(n) = n^2 \log^2 n$ $K > \log_b a$
$T(n) = n^2 \log^2 n$

⑲ $f(n) = \frac{n^3}{\log n}$ ignore $\log n$
$T(n) = n^3$

EXAMPLES

① $T(n) = 2T\left(\frac{n}{2}\right) + 1$ $\log_2 2 = 1 > 0$ @C1
then $T(n) = O(n^1)$ $O(n^{\log_b a})$

② $T(n) = 4T\left(\frac{n}{2}\right) + 1$ $\log_2 4 = 2 > 0$ @C1
$T(n) = O(n^2)$

③ $T(n) = 4T\left(\frac{n}{2}\right) + n$ $\log_2 4 = 2 > 1$ @C1
$T(n) = O(n^2)$

④ $T(n) = 8T\left(\frac{n}{2}\right) + n^2$ $\log_2 8 = 3 > 2$ @C1
$T(n) = O(n^3)$

⑤ $T(n) = T\left(\frac{n}{2}\right) + n$ $\log_2 1 < 1$ @C3
$T(n) = O(n)$

⑥ $T(n) = 2T\left(\frac{n}{2}\right) + n^2$ $\log_2 2 = 1 < 2$
$T(n) = O(n^2)$

* $f(n)$ as such
* $\log^{-ve} n$ ignore

⑦ $T(n) = 2T\left(\frac{n}{2}\right) + n^2 \log n$
$T(n) = O(n^2 \log n)$

⑧ $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n^2}{\log n}$ $\Rightarrow T(n) = O(n^2)$

(9) $T(n) = T(\frac{n}{2}) + 1$ $\log_2 1 = 0 = k$ @ C3

$T(n) = O(1 \times \log n)$

$f(n) \times \log n$ $P > -1$

(10) $T(n) = 2T(\frac{n}{2}) + n$ $f(n) \log\log n$ $P = -1$

$T(n) = O(n \log n)$ $f(n)$ $P < -1$

(11) $T(n) = 2T(\frac{n}{2}) + n\log n$

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

(12) $T(n) = 4T(\frac{n}{2}) + n^2$

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

(13) $T(n) = 4T(\frac{n}{2}) + (n \log n)^2$ $k = 2$

$T(n) = O(n^2 \log^3 n)$

(14) $T(n) = 4T(\frac{n}{2}) + \frac{n^2}{\log n}$

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

(15) $T(n) = 4T(\frac{n}{2}) + \frac{n^2}{\log^2 n}$

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

ROOT FUNCTION:

$$T(n) = \begin{cases} 1 & n = 2 \\ T(\sqrt{n}) + 1 & n > 2 \end{cases}$$

if $(n > 2)$

$T(n) = T(\sqrt{n}) + 1$

$T(n) = T(n^{1/2}) + 1$

$= T(n^{1/2^2}) + 2$

$= T(n^{1/2^3}) + 3$

$\vdots$

$T(n) = T(n^{1/2^k}) + k$

We assume, $n = 2^m$

$n^{\frac{1}{2^k}} = \frac{1}{}$

$T(2^m) = T(2^{m/2^k}) + k$

Assume $T(2^{m/2^k}) = T(2)$

$m = 2^k$

$k = \log m$

$m = \log n$

$\therefore n = \log \log_2 n$