Design Algorithm Of Analysis

Theory Assignment -)

① What is an algorithm? explain various characteristics of an algorithm.

(A) Algorithm is Nothing but step by step process for solving a problem.

* Abstract computation procedure which takes some values as input and produce a value(s) as output.
* We will write an algorithm at design time.
* It is independent of platform.
* It depends on domain knowledge.
* We analyse the algorithm function of the time
* It is an priori testing

Characteristics of an algorithm:

1) Input : The algorithm must have 0 (or) More inputs.
2) output : It must have atleast one output
3) Finiteness: It must terminate after a finite no. of steps.
4) Definiteness: Each step of algorithm must be clear
5) Effectiveness : Each step of algorithm must be correct and it should happen in finite count of time.

③ List and explain various ways of expressing language an algorithm.

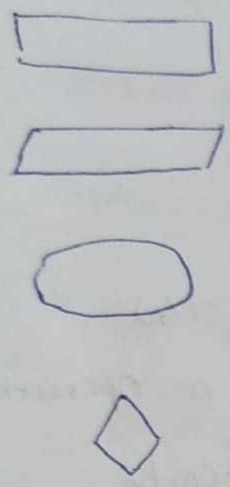
Ⓐ There are three ways of expressing an Algorithm.

1) Natural language
2) pseudo code
3) Flow chart

Natural language: It is low level language which is platform independent and can be written in simple english but it has many drawbacks associated with it like unambiguous, do not have proper structure and dificult to fund its location changes

Pseudo code: It uses programing constructs to write an algorithm and it is an platform independent The pseudo code has an advantage of being easilly converted into any programming Language This pseudo code is language independent and it has no standard of writing it.

Flow chart: Flow chart is an pictorial representation of an algorithm. It is easy to understand a flow chart. Flow chart has a simple geometric shapes to depict processes and arrows to relationships and dataflow. some Geometric shapes include



→ Rectangle → steps of algorithm

→ parallelogram → Input and output

→ oval → start / end.

→ Rhombus → condition/ selection.

3) What is amortized analysis? Explain in detail.

A) 1) Amortized analysis is a Method for analysing a given algorithms. complexity or how much resource (time or memory), it takes to excecute

2) If one operation effecting the cost on another operations in the sequence then we have to use amortized analysis, instead of worst case analysis

3) The basic idea is that worst-case operation can alter the state in such a way that the worst case cannot occur again for a long time, thus "amortizing" its cost.

(4) There are generally 3 Methods for performing amortized analysis.

a) aggregate method.

b) accounting Method.

c) potential Method.

a) Aggrigate Method: determines the upperbound T(n) on the total cost of the sequence of n operations, then calculate the amortized cost to be T(n)/n.

b) Accounting Method: It is a form of aggregate analysis which analysis to each operations an amortized cost which may differ from actual cost early operations have amortised cost greater than actual cost but as sequence of operations increases and non negative credit increases, then actual cost becomes amortized cost - accedential credit. so amortized cost is an upper bound over actual cost.

c) potential method: It is a form of accounting method where credit was saved is computed as a function (potential) of the state of the data structure.

Here amortized cost = intermediate cost + change in potential.

(4) What is probabilistic analysis of algorithm?

A) In the probabilistic analysis of algorithms we make use of probability theory. Distribution like PDF/PMF, expected value/mean i.e., variance analyse the running time of algorithms.

* It takes an assumption about a probabilistic distribution of the set of all possible inputs.
* We use it to analyse average case complexity
* In probabilistic analysis of probabilistic ~~analysis~~ algorithms the distribution or average of all possible choices in addition to the input distribution as all the algorithms do not go till the extent of using wort time complexity.
* we need to understand the distribution of inputs. or atleast make some assumpton (like uniform random general) about it.

Eg: Hiring process.
* worst case is not effective i.e, sending persons in increasing order of capability.
* Best case you to not know until you interview all.

⑤ What is a recurrance relation? How to solve recurance relation?

Ⓐ <u>Recurance Relation</u>: When an algorithm contains a recursive call to itself then its time complexity can be described by recurence relation.

There are three methods to solve recurance relations:

1) Substitution Method (Forward and backward sub^n)
2) Recurrance tree Method.
3) Master method.

<u>Substitution Method</u>:

This method has two types forward substitution and backward substitution.

<u>Example</u>:

① Find the boundry in the first step.

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

② Now Start with T(n) and expand it recursively using recurrance.

$T(n) = T(n-1) + 1 \longrightarrow (1)$

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

Substitute (2) in (1)

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

$T(n) = T(n-2) + 2 \longrightarrow (3)$

$T(n-2) = T(n-3) + 1 \longrightarrow (4)$

Substitute (4) in (3).

$T(n) = (T(n-3) + 1) + 2$

$T(n) = T(n-3) + 3 \longrightarrow (5)$

$\vdots$

$T(n) = T(n-k) + k \longrightarrow (5)$

if $n - k = 0$.

$\Rightarrow n = k$.

Sub $k = n$ in (5)

$T(n) = T(0) + n$.

$T(n) = O(n)$.

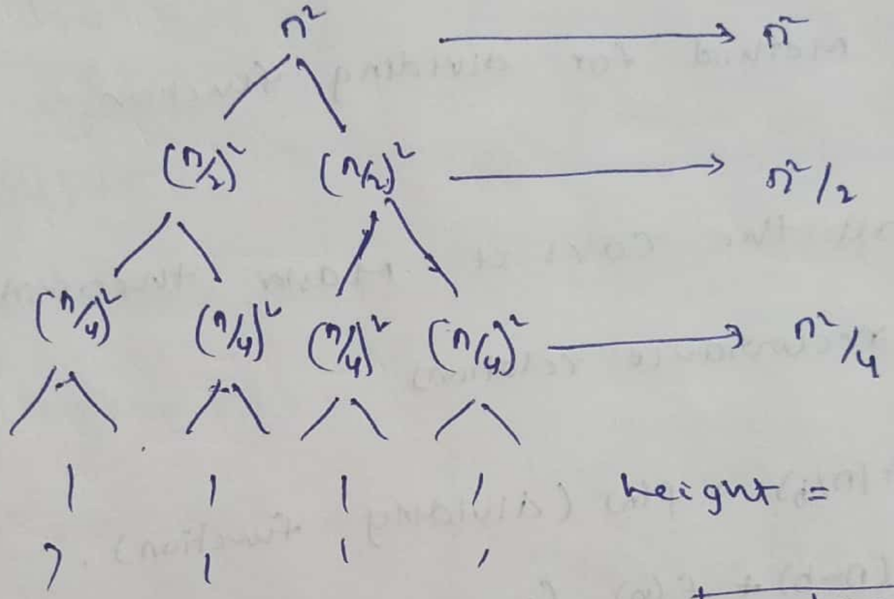
$\therefore$ The time complexity of $T(n)$ is $O(n)$.

Recurrance tree Method:

It is also known as graph Method. In recurssion tree method we should find the cost

Scanned with CamScanner

of each level, we sum the cost with each of the level of the tree to obtain a set of pre-level costs and then sum all- pre-level costs to determine the total cost of all levels of the recurssion. and then find height of the tree.



height $= \left(\frac{n}{2^k}\right)^2$

$\boxed{k = \log n.}$

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

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

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

$$T(n) = 2n^2\left(1 - (1/2)^{\log n}\right) = 2n^2\left(1 - n^{\log 1/2}\right)$$

Since we consider in growth.

$\boxed{T(n) = O(n^2).}$

Master Method:

There are two ways to solve Masters method.

1) Master Method for reducing.
(Substraction function).

2) Master Method for dividing function.

⑥ Explain all the cases of Master theorem for the following reccurance relations

$T(n) = aT(n/b) + f(n)$ (dividing function).

$T(n) = T(n-b) + f(n)$ (decreasing function).

Decreasing function:

$$T(n) = aT(n-b) + f(n).$$

Where, $a>0$, $b>0$, $f(n) = O(n^k)$.

if $a=1$, $T(n) = O(n * f(n))$

$a>1$, $T(n) = O(a^n * f(n))$

$a<1$, $T(n) = O(f(n))$.

Dividing function:

$T(n) = aT(n/b) + f(n)$.

Where, $a >= 1$, $b > 1$, $f(n) = O(n^k \log^p n)$

Case 1: if $\log_b^a > k$, then $O(n^{\log_b^a})$

Case 2: if $\log_b^a = k$.

(i) if $p > -1$ them $O(n^k \log^{p+1} n)$

(ii) if $p = -1$ then $O(n^k \log \log n)$

(iii) if $p < -1$ then $O(n^k)$.

Case 3: if $\log_b^a < k$

(i) if $p >= 0$ $O(n^k \log^p n)$

(ii) if $p < 0$ $O(n^k)$

⑦ Solve the following Recurrence relations using Substitution, recurrence tree and Master method.

① $T(n) = 7T(n/2) + 3n^2 + 2$

$$T(n) = \begin{cases} 1 & , \text{ if } n=1 \\ 7T(n/2) + 3n^2 + 2 & , \text{ otherwise} \end{cases}$$

$T(n/2) = 7T(n/4) + 3(n/2)^2 + 2$

$T(n) = 7(7T(n/4) + 3(n/2)^2 + 2) + 3n^2 + 2$

$= 7^2 T(n/4) + 7.3.(n/2)^2 + 7.2 + 3n^2 + 2$

$T(n) = 7^k T(n/2^k) + 3n^2 \left[1 + \frac{7}{4} + \frac{7^2}{4^2} + \cdots \right] + 2[1 + 7 + 7^2 + \cdots]$

$T(n) = 7^k T(n/2^k) + 3n^2 \left(\frac{(7/4)^k - 1}{(7/4) - 1} \right) + 2\left(\frac{7^k - 1}{7 - 1} \right)$

if $n/2^k = 1 \Rightarrow \boxed{k = \log n}$

$T(n) = 7^{\log n} T(n/2^k) + 3n^2 \left(\frac{(7/4)^{\log n} - 1}{7/4 - 1} \right) + 2\left(\frac{7^{\log n} - 1}{6} \right)$

$T(n) = n^{\log 7}(1) + 4n^2 ((7/4)^{\log n} - 1) + \left(\frac{n^{\log 7} - 1}{3} \right)$

$T(n) = n^2 . n^{\log 7/4}$

$T(n) = n^{2 + \log 7 - \log 4}$

$\boxed{T(n) = O(n^{\log 7})}$

Scanned with CamScanner

② $T(n) = 2T(n/2) + n^2$

$a = 2,\ b = 2,\ f(n) = n^2,\ k = 2,\ p = 0$

$\log_b a = \log_2 2 = 1 \Rightarrow 1 < k$

$\because \log_b a < k$ and $p = 0$

$T(n) = O(n^k \log^p n)$

$\boxed{T(n) = O(n^2)}$

④ $T(n) = 4T(n/2) + n^3$

$a = 4,\ b = 2,\ f(n) = n^3,\ p = 0,\ k = 3$

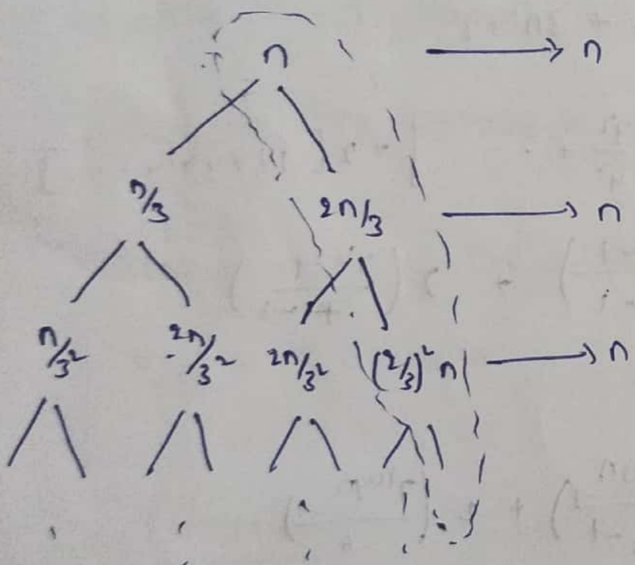
$\log_b a = \log_2 4 = 2 < k$

$\because \log_b a < k$ & $p = 0$

$T(n) = O(n^k \log^p n)$

$\boxed{T(n) = O(n^3)}$

③ $T(n) = T(n/3) + T(2n/3) + n$



$n,\ 2n/3,\ (\frac{2}{3})^2 n,\ \dots,\ (\frac{2}{3})^k n.$

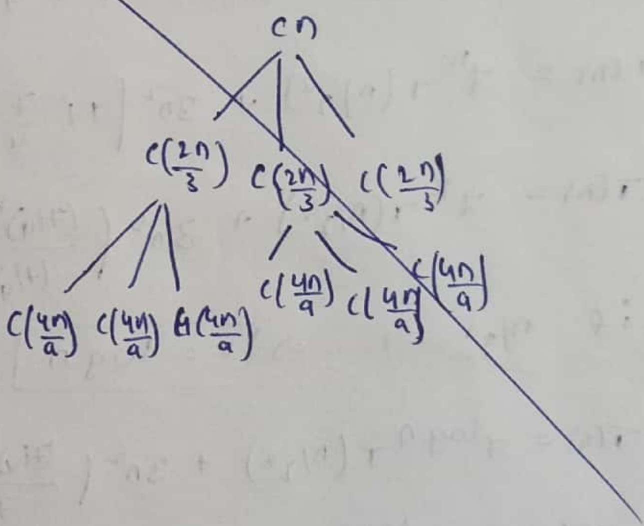
if, $(\frac{2}{3})^k n = 1 \Rightarrow n = (2/3)^{-k}$

$\Rightarrow \boxed{k = \log_{3/2} n}$

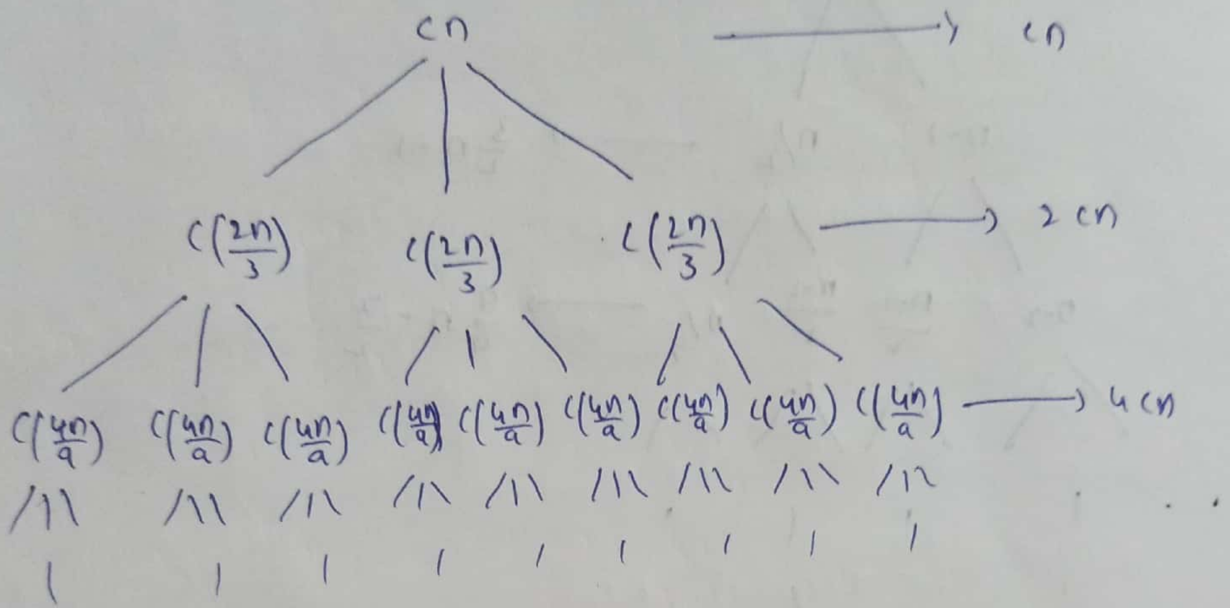
$T(n) = kn$

$\boxed{T(n) = O(n \log_{3/2} n)}$

⑤ $T(n) = 3T(2n/3) + cn$



(5) $T(n) = 3T(2n/3) + cn$



$cn, \frac{2}{3}cn, \frac{4}{9}cn, \ldots \left(\frac{2}{3}\right)^k \cdot cn$

if $\left(\frac{2}{3}\right)^k cn = 1$

$cn = \left(\frac{3}{2}\right)^k$

$\log cn = k \log 3/2$

$\boxed{k = \log_{3/2} cn}$

$T(n) = cn + 2cn + 4cn + \ldots + 2^k cn$

$T(n) = cn(1 + 2 + + \ldots + 2^k)$

$T(n) = cn\left(\frac{1(2^k - 1)}{2-1}\right)$

$T(n) = cn(2k - 1)$

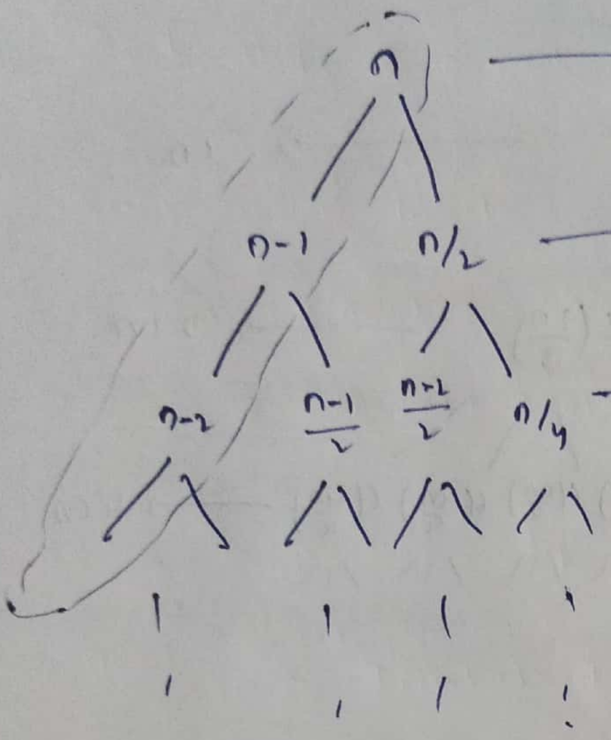
$T(n) = cn\, 2^{\log_{3/2} cn} - cn$

$T(n) = \cancel{cn \log^{\log_{3/2} 2}}$

$T(n) = (cn)(n^{\log_{3/2} 2}) - cn$

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

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



$\rightarrow n$

$\rightarrow \frac{3}{2}n - 1$

$\rightarrow \frac{9}{4}n - \frac{7}{2}$

$n, n-1, n-2, \ldots, n-k$

if $n-k=0$.

$\boxed{n=k.}$

Total cost $= T(n) = n + \frac{3}{2}n - 1 + \frac{9}{4} - \frac{7}{2} + \ldots +$

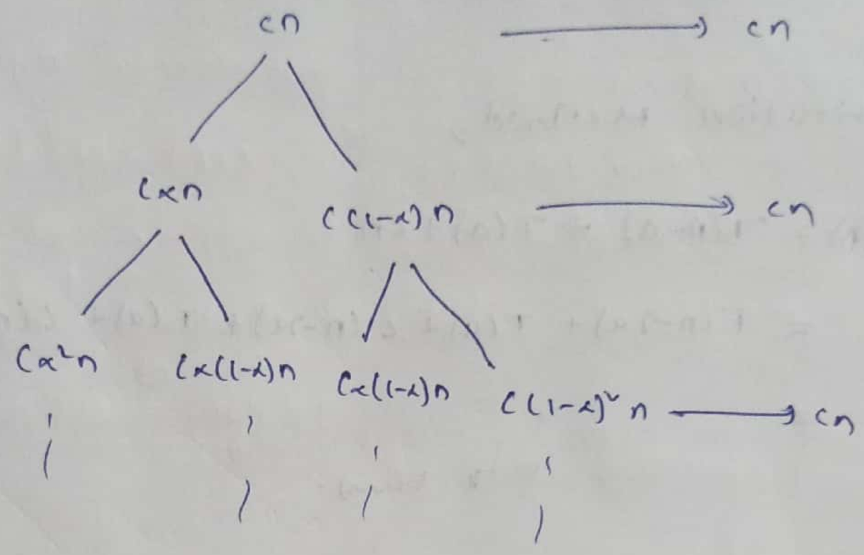
$= n + \frac{3n}{2} - 1 + \frac{9n}{4} - \frac{7}{2} + \ldots + 2^k.$

$= n\left[1 + \frac{3}{2} + \ldots + \left(\frac{3}{2}\right)^k\right] + 2^k.$

$= n(\text{cost}) + 2^n$

$\boxed{T(n) = O(2^n).}$

(7) $T(n) = T(\alpha n) + T((1-\alpha)n) + cn.$



$cn,\ c(1-\alpha)n,\ c(1-\alpha)^2 n, \ldots, c(1-\alpha)^k n.$

$c(1-\alpha)^k n = 1$

$\log c(1-\alpha)^k + \log n = 0$

$\log cn + k \log(1-\alpha) = 0.$

$\log cn = -k \log(1-\alpha)$

$\boxed{k = \log_{cn}(1-\alpha)}$

$T(n) = k(cn)$

$T(n) = (c)\, n \log_{cn}^{1-\alpha}$

$\boxed{T(n) = O(n \log_{cn}^{(1-\alpha)})}$

(8) $T(n) = T(n-a) + T(a) + cn$.

$a \geq 1$ and $c > 0$ are constants.

Substitution Method,

$T(n) = T(n-a) + T(a) + cn$.

$= T(n-2a) + T(a) + c(n-2a) + T(a) + c(n-a) + T(a) + cn$.

$=$

k times.

$T(n-ka) = T(0)$

$n = ka$

$\boxed{k = n/a}$

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

$= O(n^2)$

$\boxed{T(n) = O(n^2).}$

Scanned with CamScanner

(9) $T(n) = 2T(n/4) + 1$

$a = 2, b = 4, f(n) = 1, k = 0, p = 0$

$\log_b a = \log_4 2 = 1/2 > k$

$\because \log_b a > k$ and $p = 0$

$T(n) = O(n^{\log_b a})$

$T(n) = O(n^{\log_4 2})$

$\boxed{T(n) = O(\sqrt{n})}$

(10) $T(n) = 2T(n/4) + \sqrt{n}$

$a = 2, b = 4, f(n) = \sqrt{n}, k = 1/2, p = 0$

$\log_b a = \log_4 2 = 1/2 = k$

$\because \log_b a = \log_4 2 = 1/2 = k$ and $p = 0$

$T(n) = O(n^k \log^{p+1} n)$

$T(n) = O(n^{1/2} \log^{0+1} n)$

$\boxed{T(n) = O(\sqrt{n} \log n)}$

(11) $T(n) = 2T(n/4) + n$

$a = 2, b = 4, f(n) = n, k = 1, p = 0$

$\log_b a = \log_4 2 = 1/2 < k$

$T(n) = O(n^k \log^p n)$

$T(n) = O(n^{1} \log^0 n)$

$\boxed{T(n) = O(n)}$

(12) $T(n) = 2T(n/4) + n^2$

$a = 2, b = 4, f(n) = n^2, k = 2, p = 0$

$\log_b a = \log_4 2 = 1/2 < k$

$\because \log_b a < k$ and $p = 0$

$T(n) = O(n^k \log^p n)$

$T(n) = O(n^2 \log^0 n)$

$\boxed{T(n) = O(n^2)}$

(13) $T(n) = 4T(n/2) + n^2 \log n$

$a = 4, b = 2, f(n) = n^2 \log n, p = 1, k = 2$

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

$\because \log_b a = k$ and $p = 1$

$T(n) = O(n^k \log^{p+1} n)$

$\boxed{T(n) = O(n^2 \log^2 n)}$

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

$a = 1, a > 0, b > 0, f(n) = n$

$\because a = 1,$

$T(n) = O(n * f(n))$

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

$\boxed{T(n) = O(n^2)}$

(15) $T(n) = T(\lceil n/2 \rceil) + 1$

Asymptotic solutions are unaffected by ceil function

so, $T(n) = T(\lceil n/2 \rceil + 1)$ can be written as $T(n) = T(n/2) + 1$

$a=1, b=2, k=0, p=0, f(n)=1$

$\log_b a = 0 = k.$

$\therefore T(n) = O(n^k \log^{p+1} n)$

$\boxed{T(n) = O(\log n)}$

(16) $T(n) = 2T(\lfloor n/2 \rfloor) + n$

Asymptotic solutions are unaffected by floor function.

so, $T(n) = 2T(\lfloor n/2 \rfloor) + n$. can be written as $T(n) = 2T(n/2) + n$.

$a=2, b=2, k=1, p=0, f(n)=n$

$\log_b a = \log_2 2 = 1 = k.$

$\therefore T(n) = O(n^k \log^{p+1} n)$

$\boxed{T(n) = O(n \log n)}$

(17) $T(n) = 4T(n/2) + n^2;$

$a=4, b=2, f(n)=n^2, k=2, p=0$

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

$\because \log_b a = k$ and $p > -1$

$T(n) = O(n^k \log^{p+1} n)$

$T(n) = O(n^2 \log^{0+1} n)$

$\boxed{T(n) = O(n^2 \log n)}$

(18)

(19) $T(n) = 2T(n/2) + n^4$

$a=2, b=2, k=4, p=0, f(n)=n^4$

$\log_b a = \log_2 2 = 1 < k.$

$\because \log_b a < k$ and $p=0$

$T(n) = O(n^k \log^p n)$

$T(n) = O(n^4 \log^0 n)$

$\boxed{T(n) = O(n^4)}$

(20) $T(n) = T(7n/10) + n$

$a=1, b=10/7, f(n)=n, k=1, p=0$

$\log_b a = \log_{10/7} 1 = 0 < k$

$T(n) = O(n^k \log^p n)$

$T(n) = O(n^1 \log^0 n)$

$\boxed{T(n) = O(n^1)}$

(21) $T(n) = 16T(n/4) + n^2$

$a=16, b=4, f(n)=n^2, k=2, p=0$

$\log_b a = \log_4 16 = 2 = k$

$\because \log_b a = k$ and $p > -1$

$T(n) = O(n^k \log^{p+1} n)$

$T(n) = O(n^2 \log^{0+1} n)$

$\boxed{T(n) = O(n^2 \log n)}$

Scanned with CamScanner

(18) $T(n) = 3T(\sqrt{n}) + \log n$

$$T(n) = \begin{cases} 1, & \text{if } n \le 2 \\ 3T(\sqrt{n}) + \log n, & \text{otherwise} \end{cases}$$

$T(\sqrt{n}) = 3T(n^{1/4}) + \log(\sqrt{n})$

$T(n) = 3(3T(n^{1/4}) + \log\sqrt{n}) + \log n.$

$T(n) = 3^2 T(n^{1/4}) + 3\log\sqrt{n} + \log n$

$T(n) = 3^k T(n^{1/2^k}) + \log n \cdot n^{3/2} \cdot n^{3^2/4} \cdots n^{3^k/2^k}$

$T(n) = 3^k T(n^{1/2^k}) + (1 + 3/2 + 3^2/4 + \cdots + 3^k/2^k)\log n.$

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

$T(n) = 3^k \cdot T(n^{1/2^k}) + 2(3/2)^k \log n - 2\log n.$

if $n^{1/2^k} = 2$

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

$\Rightarrow \boxed{\log(\log n) = k}$

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

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

$\boxed{T(n) = O((\log n)^{1+\log 3/2})}$. ($\because$ we consider growth).

(22) $T(n) = 7T(n/3) + n^2$

$a=7,\ b=3,\ k=2,\ f(n)=n^2,\ p=0$

$\log_3 7 = 1.771 < k$

$\because \log_b a < k$ and $p=0$

$T(n) = O(n^k \log^p n)$

$T(n) = O(n^2 \log^0 n)$

$\boxed{T(n) = O(n^2)}$

(23) $T(n) = 7T(n/2) + n^2$

$a=7,\ b=2,\ f(n)=n^2,\ k=2,\ p=0$

$\log_b a = \log_2 7 = 2.8 > k$

$\because \log_b a > k$ and $p=0$

~~$T(n) = O(n^k \log^p n)$~~ $T(n) = O(n^{\log_b a})$

~~$T(n) = O(n^2 \log^0 n)$~~ $T(n) = O(n^{\log_2 7})$

~~$\boxed{T(n) = O(n^2)}$~~ $\boxed{T(n) = O(n^{\log_2 7})}$

(24) $T(n) = 2T(n/4) + \sqrt{n}$

$a=2,\ b=4,\ f(n)=\sqrt{n},\ k=1/2,\ p=0$

$\log_b a = \log_4 2 = 1/2 = k$

$\because \log_b a = k$ and $p > -1$

$T(n) = O(n^k \log^{p+1} n)$

$T(n) = O(n^{1/2} \log^1 n)$

$\boxed{T(n) = O(\sqrt{n})}$

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

$a=1,\ f(n)=n^2$

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

$\boxed{T(n) = O(n^3)}$

⑧ Group the functions so, that $f(n)$ and $g(n)$ are in same group if $f(n) = O(g(n))$ and $g(n) = O(f(n))$. List the group in increasing order.

$\sqrt{n}$	n	2^n
$n\log n$	$n - n^3 + 7n^5$	$n^2 + \log n$
n	n^3	$\log n$
$n^{1/3} + \log n$	$\log^2 n$	$n!$
$\ln n$	$n/\log n$	$\log\log n$
$(1/3)^n$	$(2/3)^n$	6

⑨ Time complexities:

$\sqrt{n} = O(\sqrt{n})$

$n\log n = O(n\log n)$

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

$n^{1/3} + \log n = O(n^{1/3})$

$\ln n = O(\log n)$

$(1/3)^n = O((1/3)^n)$

$n = O(n)$

$n - n^3 + 7n^5 = O(n^5)$

$n^3 = O(n^3)$

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

$(3/2)^n = O((3/2)^n)$

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

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

$\log n = O(\log n)$

$n! = O(n!)$

$\log\log n = O(\log\log n)$

$6 = O(1)$

$n/\log n = O(n/\log n)$

$O(1) \longrightarrow 6$

$O(\log\log n) \longrightarrow \log(\log n)$

$O(\log n) \longrightarrow \log n$

$\ln n \quad (\because \ln n \sim \log n)$

$O(\log^2 n) \longrightarrow \log^2 n$

$O((1/3)^n) \longrightarrow (1/3)^n$

$O(n^{1/3}) \longrightarrow n^{1/3} + \log n$

$O(n^{1/2}) \longrightarrow \sqrt{n}$

$O(n/\log n) \longrightarrow n/\log n$

$O(n) \longrightarrow n$

$O(n\log n) \longrightarrow n\log n$

$O(n^2) \longrightarrow n^2$

$n^2 + \log n \ldots$

$O(n^3) \longrightarrow n^3$

$O(n^5) \longrightarrow n^5$

$\cancel{O((1/3)^n) \longrightarrow (1/3)^n}$

$O((3/2)^n) \longrightarrow 2^n$

$O(n!) \longrightarrow n!$

Increasing order.

Order of time complexity:

$O(1) < O(\log\log n) < O(\log n) < O(\log^2 n) < O((1/3)^n) < O(n^{1/3}) < O(\sqrt{n})$
$< O(n/\log n) < O(n) < O(n\log n) < O(n^2) < O(n^3) < O(n^5) < \cancel{O((1/3)^n)}$
$< O((3/2)^n) < O(2^n) < O(n!)$

(9) Relative asymptotic growths.

Indicate, for each pair of expressions (A, B) in table below, weather A is O, o, Ω, ω, Θ of B. Assume $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of the table with "yes" or "NO" written in each box.

A)

A	B	O	o	Ω	ω	Θ
$\log^k n$	n^ϵ	yes	yes	no	no	no
n^k	c^n	yes	yes	no	no	no
$\sqrt{n}$	$n^{\sin n}$	no	no	no	no	no
2^n	$2^{n/2}$	no	no	yes	no	yes
$n^{\lg c}$	$c^{\lg n}$	yes	no	yes	no	yes
$\lg(n!)$	$\lg(n^n)$	yes	no	yes	no	yes.

(10) Consider the following fourteen functions for the questions that follows.

(a) $\log_3(2n)$

(b) $\sqrt{n}$

(c) $n\log_3(n/2)$

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

(e) 2^n

(f) 2^{n+2}

(g) 2^{2n}

(h) $3n+5\log_2(n)$

(i) $5n+\sqrt{n}$.

(j) $\sum_{k=1}^{n} k = 1+2+3+\cdots+n$

(k) $\sum_{k=1}^{2n} k = 1+2+3+4+\cdots+2n$

(l) $\sum_{k=1}^{n^2} k = 1+2+3+4+\cdots+n^2$

(m) $\sum_{k=1}^{n} k^3 = 1^3+2^3+3^3+\cdots+n^3$.

(n) $\sum_{k=1}^{n} 2^k = 1+2+4+8+\cdots+2^n$.

Make a table in which each function is a column directed by its Θ growth rate. Function with the same asymptotic growth rate should be in same column. Columns should be ordered left to right by the rate of growth of their functions: colums with slower growing functions should to be left of columns with faster growing functions.

$< O(3/2^n) < O(2^n) < O(n!)$

(10) (A)

(a) $\log_3 2n = \log_3 2 + \log_3 n = O(\log n)$

(b) $\sqrt{n} = O(\sqrt{n})$

(c) $n \log_3 n/2 = O(n \log n)$.

(d) $\log_2 3n^2 = 2 \log_2 3n = 2 \log_2 3 + 2 \log_2 n = O(n \log n)$

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

(f) $2^{n+2} = O(2^n)$

(g) $2^{2n} = 4^n$

(h) $3n + 5 \log_2 n = O(n)$

(i) $5n + \sqrt{n} = O(n)$

(j) $\sum_{k=1}^{n} k = 1 + 2 + 3 + \dots + n = n(n+1)/2 = O(n^2)$

(k) $\sum_{k=1}^{2n} k = 1 + 2 + 3 + \dots + 2n = n(n+1)/2 = O(n^2)$

(l) $\sum_{k=1}^{n^2} k = 1 + 2 + 3 + \dots + n^2 = n(n+1)/2 = O(n^2)$.

(m) $\sum_{k=1}^{n} k^3 = 1^3 + 2^3 + 3^3 + \dots + n^3 = n^2(n^2+1)/2 = O(n^4)$.

(n) $\sum_{k=1}^{n} 2^k = 2^0 + 2^1 + \dots + 2^k = 1(2^n - 1)/2-1 = O(2^n)$.

$O(\log n) \longrightarrow \log_3 2n$

$\log_{\frac{3}{2}} n^2$

$O(\sqrt{n}) \longrightarrow \sqrt{n}$

$O(n) \longrightarrow 3n + 5\log_2(n)$

$5n + \sqrt{n}$

$O(n \log n) \longrightarrow n \log_2 n/2$

$O(n^2) \longrightarrow \sum_{k=1}^{n} k, \ \sum_{k=1}^{2n} k, \ \sum_{k=1}^{n^2} k$

Increasing order.

$O(n^4) \longrightarrow \sum_{k=1}^{n} k^3$

$O(2^n) \longrightarrow \sum_{k=1}^{n} 2^k,$

$2^n, \ 2^{n+2}.$

$O(4^n) \longrightarrow 2^{2n}.$

$\therefore$ Order of Time complexity:

$O(\log n) < O(\sqrt{n}) < O(n) < O(n \log n) < O(n^2) < O(n^4) < O(2^n) < O(4^n)$

(11) For each of the following pair of functions, either $f(n)$ is $O(g(n))$, $f(n)$ is $\Omega(g(n))$, or $f(n)$ is $\Theta(g(n))$. Determine which relationship is corrected and explain.

a) $f(n) = n^{0.25}$ $\quad g(n) = n^{0.5}$

$f(n) = \Theta(n^{1/4})$, $g(n) = \Theta(n^{1/2})$

$f(n) = O(g(n))$

because $g(n)$ curve is greater than curve of $f(n)$.

for values n grater than 1.

b) $f(n) = n$, $g(n) = \log^2 n$.

$f(n) = \Omega(\log^2 n)$

$\boxed{f(n) = \Omega(g(n))}$

because $g(n)$ curve is below the curve of $f(n)$.

c) $f(n) = \log n$, $g(n) = \ln n$.

$f(n) = \Theta(\log n)$, $g(n) = \Theta(\log n)$

$f(n) = \Theta(g(n))$.

Since the growth is same for $f(n)$ and $g(n)$.

d) $f(n) = 1000n^2$, $g(n) = 0.0002n^2 - 1000n$.

$f(n) = \Theta(n^2)$, $g(n) = \Theta(n^2)$.

$f(n) = \Theta(g(n))$.

since the growth is same for $f(n)$ and $g(n)$.

e) $f(n) = n\log n$, $g(n) = n\sqrt{n}$

$f(n) = \Theta(n\log n)$, $gn = \Theta(n\sqrt{n})$.

$f(n) = O(g(n))$.

Since, $g(n)$ curve is above $f(n)$ curve. so, $f(n) = O(g(n))$.

f) $f(n) = e^n$, $g(n) = 3^n$.

$f(n) = \Theta(e^n)$, $g(n) = \Theta(3^n)$

$f(n) = \Omega(g(n))$.

Since, $g(n)$ curve is below $f(n)$ curve. So, $f(n) = \Omega(g(n))$

g) $f(n) = 2^n$, $g(n) = 2^{n+1}$

$f(n) = \Theta(2^n)$, $g(n) = \Theta(2^n)$.

$f(n) = \Theta(g(n))$.

Since, the grown is same for $f(n)$ and $g(n)$.

So, $f(n) = \Theta(g(n))$.

h) $f(n) = 2^n$, $g(n) = 2^{2n}$

$f(n) = \Theta(2^n)$, $g(n) = \Theta(4^n)$

$f(n) = O(g(n))$

Since, $f(n)$ curve is below the curve $g(n)$.

so, $f(n) = O(g(n))$.

i) $f(n) = 2^n$, $g(n) = n!$

$f(n) = \theta(2^n)$, $g(n) = \theta(n!)$

$f(n) = O(g(n))$.

Since, $g(n)$ curve is above the curve $f(n)$ so,

$f(n) = O(g(n))$.

j) $f(n) = \lg n$, $g(n) = \sqrt{n}$.

$f(n) = \theta(\log n)$, $g(n) = \theta(\sqrt{n})$.

$f(n) = O(g(n))$.

Since, the curve $g(n)$ is above the curve $f(n)$. so,

$f(n) = O(g(n))$.

k) $f(n) = \log n^2$, $g(n) = \log n + 5$.

$f(n) = \theta(\log n)$, $g(n) = \theta(\log n)$

$f(n) = \theta(g(n))$.

Since, the growth is same for $f(n)$ and $g(n)$.

so, $f(n) = \theta(g(n))$.

l) $f(n) = n$, $g(n) = \log n^2$.

$f(n) = \theta(n)$, $g(n) = \theta(\log n)$

$f(n) = \Omega(g(n))$.

since, the curve $g(n)$ is below the curve $f(n)$, so

$f(n) = \Omega(g(n))$.

m) $f(n) = \log\log n$, $g(n) = \log n$.

$f(n) = \Theta(\log\log n)$, $g(n) = \Theta(\log n)$.

$f(n) = O(\log n)$.

Since, the curve $g(n)$ is above the curve $f(n)$. So,
$f(n) = O(\log n)$.

n) $f(n) = n$, $g(n) = \log^2 n$.

$f(n) = \Theta(n)$, $g(n) = \Theta(\log^2 n)$.

$f(n) = \Omega(g(n))$.

Since, the curve $g(n)$ is below the curve $f(n)$. so,
$f(n) = \Omega(g(n))$.

~~p) f(n) = 10, g(n) = log 10.~~

o) $f(n) = n\log n + n$, $g(n) = \log n$.

$f(n) = \Theta(n\log n)$, $g(n) = \Theta(\log n)$.

$f(n) = \Omega(g(n))$.

Since, $g(n)$ is below the curve $f(n)$. So, $f(n) = \Omega(g(n))$.

p) $f(n) = 10$, $g(n) = \log 10$.

$f(n) = \Theta(1)$, $g(n) = \Theta(1)$

$f(n) = \Theta(g(n))$.

Since, the growths are same $f(n) = \Theta(g(n))$.

a) $f(n) = 2^n$, $g(n) = 10n^2$

$f(n) = \Theta(2^n)$, $g(n) = \Theta(n^2)$.

$f(n) = \Omega(g(n))$.

because the ~~grow~~ curve $g(n)$ is below the curve $f(n)$

so, $f(n) = \Omega(g(n))$

b) $f(n) = 2^n$, $g(n) = 3^n$.

$f(n) = \Theta(2^n)$, $g(n) = \Theta(3^n)$.

$f(n) = O(g(n))$.

since, the curve $g(n)$ is above the curve $f(n)$. so,

$f(n) = O(g(n))$.