ANALYTICAL

QUESTIONS

CSA0666 - DESIGN AND

ANALYSIS OF ALGORITHMS

NAME : TEJASWINI . P

REGNO : 192324046.

192324046

1) Solve the following recurrence relation.

a) $x(n) = x(n-1)+5$ for $n>1$ $x(1)=0$.

$x(n) = x(n-1)+5$

$x(1) = 0$

subs $n=2$

$x(2) = x(2-1)+5 = 0+5 = 5$

$n=3$

$x(3) = x(3-1)+5$

$= x(2)+5 = 5+5 = 10$

$x(3) = 10$

$n=4$

$x(4) = x(3)+5 = 10+5 = 15$

$n=5$:-

$x(5) = x(5-1)+5 = 15+5 = 20$

$x(n) = x(n-1)+5$

$= x(1) + 5(n-1)$

$= x(1) + 5(n-1)$

$= 0 + 5(n-1)$

$= 5(n-1)$

$$\boxed{x(n) = 5(n-1)}$$

b) $x(n) = 3x(n-1)$ for $n > 1$ $x(1) = 4$.

$x(n) = 3x(n-1)$

$x(1) = 4$

<u>$n = 2$</u>

$x(2) = 3x(2-1)$

$= 12$

$n = 3$

$x(3) = 3x(3-1)$

$= 36$

$n = 4$ $x(4) = 3x(3) = 108$

$n = 5$ $x(5) = 3x(4) = 324$.

Now, $x(n) = 3x(n-1)$

$x(n) = 3^{n-1} \cdot x^{(1)}$

$= 3^{n-1} \cdot 4$

$\boxed{x(n) = 4 \cdot 3^{n-1}}$

c) $x(n) = x(n/2) + n$ for $n > 1$ $x(1) = 1$, $n = 2k$

$x(n) = x(n/2) + n$, $x(1) = 1$

$n = 2^1 = 2$

$n = 2^2 = 4$

$n = 2^3 = 8$.

$n = 2$

$x(2) = x(2/2) + 2$

$= 1 + 2 = 3$

$n = 4$ $x(4) = x(2) + \cancel{2} = 3 + \cancel{2}4 = 7$

$n = 8$

$x(8) = x(8/2) + 8$

$= 7 + 8$

$x(8) = 15.$

so, $\boxed{x(2^k) = 2^{k+1} - 1}$

d) $x(n) = x(n/3) + 1$ for $n > 1$ $x(1) = 1$, $n = 3k$

$x(n) = x(n/3) + 1$, $x(1) = 1.$, $n = 3k.$

$n = 3^1 = 3$

$n = 3^2 = 9$

$n = 3^3 = 27.$

$x(3) = x(3/3) + 1$

$= x(1) + 1$

$x(3) = 2$

$x(9) = x(9/3) + 1$

$= 2 + 1$

$x(9) = 3$

$x(27) = x(27/3) + 1$

$= 3 + 1$

$x(27) = 4.$

$\therefore$ $x(n) = \log_3 n$

$x(n) = \log 3^{3k}$

2) Evaluate the following recurrences

(i) $T(n) = T(n/2)+1$ where, $n = 2k$

$k \geq 0$.

$T(n) = T(n/2)+1$, $n = 2k$.

$T(2^k) = T(2^k/2)+1$

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

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

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

$T(2^1) = T(2^0)+1$

$k = \log_2 n$.

$T(2^k) = T(2^{k-1})+1 = T(2^{k-2})+1+1 \ldots T(2^0)+k$.

$2^0 = 1$, $T(2^0) = T(1)$

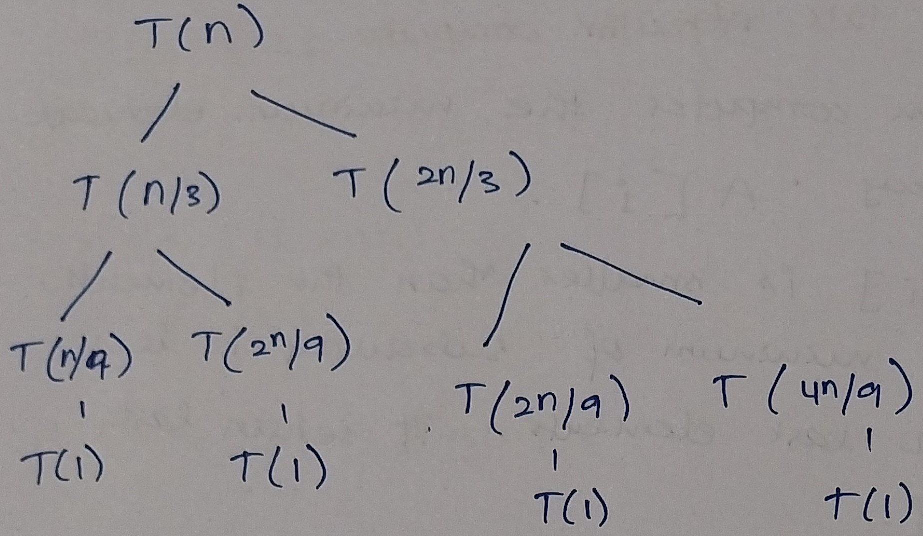
$T(2^k) = 1+k$ $\Rightarrow T(1) = 1$

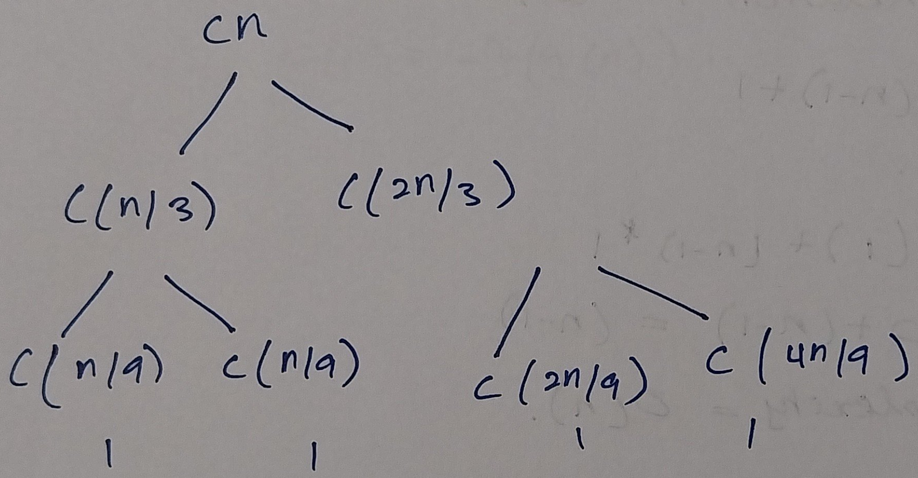
$$\boxed{T(n) = 1+\log_2 n}$$

time complexity $\Rightarrow O(\log n)$

(ii) $T(n) = T(n/3)+T(2n/3)+cn$.

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





$\log_3 n \Rightarrow$ divide by 3.

$T(n) = cn \log_3 n \Rightarrow \omega(n \log n)$.

3) consider the following recursion Algorithm.

Min1(A[0...n-1])

if n=1 return A[0]

else temp = Min1(A(0...n-2))

if temp <= A[n-1] return temp.

else

return [n-1]

a) What does this Algorithm compute

- This algorithm computes the minimum element in an array · A [i].
- i < n, A [i] is smaller than the elements.
- check the minimum of subarray is less or equal to last elements. it return last element itself.

b) recurrence relation for algorithms.

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

$T(1) = 0$.

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

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

$\therefore$ time complexity $= O(n)$.

4) (i) $F(n) = 2n^2 + 5$ & $g(n) = 7n$. use Ω g(n) notation.

condition => $F(n) \geq c \cdot g(n)$

$F(n) = 2n^2 + 5$

$c \cdot g(n) = 7n$

n = 1 $F(1) = 2(1^2) + 5 = 7$

$g(1) = 7$.

n = 2

$g(2) = 14$.

$F(n) = 2(2)^2 + 5$

$= 13$.

n = 3

$F(3) = 2(3)^2 + 5$

$= 18 + 5 = 23$

$g(n) = 27$

$n = 1, \quad 7 = 7$

$n = 2, \quad 13 = 14$

$n = 3, \quad 23 = 21$

$n \geq 3, \quad F(n) \geq g(n) \cdot c$

$F(n)$ is greater than $g(n)$

$\therefore \quad F(n) = \Omega(g(n))$.