6.08.2024.
Tuesday.

Assignment · DAA.

Saravanan Sha. S
192321127.

1. Solve The following.

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

at $n=1$; $x(1)=0$

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

$x(3) = x(3-1)+5$
$= x(2)+5$
$= 10$.

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

$x(n)$ is +5 for each increment.

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

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

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

$n=1$; $x(1) = 4$ (Given)

$n=2$; $x(2) = 3x(2-1)$
$= 3x(1)$
$= 3\times4 = 12$

$n=3$; $x(3) = 3x(3-1)$
$= 3x(2)$
$= 3\times12$
$x(3) = 36$

$n=4$; $x(4) = 3x(4-1)$
$= 3(36)$
$= 108$.

$(x(n))$ is obtained

$\therefore x(n) = 4\times3^{n-1}$

C). $x(n) = x(n/2)+n$ $n>1$ $x(1)=1$
Solve $n=2^k$]

$n=2^k$

$n=1$; $x(1)=1$

$n=2$; $x(2) = x(2/2)+n$
$= 1+2 = 3$

$n=4$; $x(4/2)+4$
$= x(2)+4$
$= 7$.

$n=8;\ n(8/2)+8$

$= n(4)+8$

$= 7+8 = 15$

$n=16;\ n(8)+16$

$= 15+16 = 31$

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

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

$$2^k = n$$

$$x(n) = x(2^k) = 2^{(\log_2 n)+1} - 1$$

$$= 2.2^{\log_2 n} = 1$$

$$= 2n-1$$

d). $x(n) = x(n/3)+1 \quad n>1 \quad x(1)=1$ Solve $n=3^k$)

$x(1) = 1$

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

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

$x(27) = x(27)+1 = 4.$

$x(n) = 1 + \log_3 n$

). Evaluate The following.

i. $T(n) = T(n/2)+1$; where $n=2k$ for all $k=0$.

Assume $n=2^k$ i.e $k-\log n$.

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

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

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

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

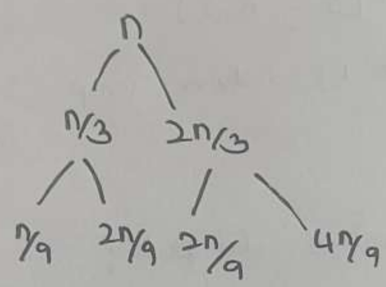
$T(2^k) = T(2^{k-k}) + k = T(2^0) + k = T(1) + k.$

$T(1) = 1$ i.e $T(n) = \log n + 1.$

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

$T(n) = (\log n)$

ii $T(n) = T(n/3) + T(2n/3) + Cn$, where c is constant.



$T(n) =$ Sum of all number.

length $= \log_3 n.$

$T(n) > n \log_{3/2} n$ $\therefore T$ is $\Omega(n \log n)$

depth $= \log_{3/2} n$

$T(n) \le n \log_{3/2} n.$

analyse The order of growth.

P. $f(n) = 2n^2+5$ & $g(n) = 7n$. use The $\Omega g(n)$.

$f(n) = 2n^2+5 >= C.(7n).$ $f(n) \geq C.g(n)$

if $n=1;\ 7 = 7$

$n=2;\ 13 = 14$

$n=3;\ 23 = 21$

$n = 4$; $37 > 28$ $f(n)$ is greater

$n = 5$, $55 > 35$ $f(n) \geq g(n)$.

3). Consider The following.

min [A [0] ... n-1].

if n-1 return A[0]

else temp = min A [0 ... n-2]

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

else

return A(n-1)

[a]. wt does This algorithm compute?

return min value in array A.

i Best case (n = 1)

if n=1, only one element. It return The A[0] and its min vale in a single element array.

2. recursive case:

if n>1, create The Temp

call recursively (A[0 to n-2]) = first n-1 element.

Comparing Temp with last element (A[n-1])

if Temp < A(n-1)

return Temp

else

return A[n-1].

8 Set up a recurrance relation for The algorithm basic operation Count & Solve it.

Base Case: $T(1) = C_1$ [Constant C_1]

recursive Case $= T(n) = T(n-1) + C_2$ [$C_2 \to$ Constant]

<u>final Solution</u>:-

$$T(n) = C_2 * n^2 + (C_1 - C_2)$$

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