Y. Harika
192325023
CSA0669

ANALYTICAL QUESTIONS - I

1) Solve the following recurrence relations

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

Given

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

$x(1) = 0$ [For $n=1$]

For $n = 2$:

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

$= x(1) + 5$

$= 0 + 5 = 5$

For $n = 3$

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

$= x(2) + 5$

$= 5 + 5 = 10$

For $n = 4$

$x(4) = x(4-1) + 5$

$= x(3) + 5$

$= 10 + 5 = 15$

For $n = 5$

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

$= x(4) + 5$

$= 15 + 5 = 20$

From the pattern we can observe:

$x(n) = x(n-1) + 5 \quad = \quad x(n) = 0 + 5(n-1)$

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

The General form for the eqn:

$x(n) = x(1) + (x-1) \cdot d$

$x(n) = 0 + (x-1) \cdot 5$

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

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

Given

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

For $n=1$:

$x(1) = 4$

For $n=2$:

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

$= 3x(1)$

$= 3 \cdot 4 = 12$

For $n = 3$

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

$= 3x(2)$

$= 3 \cdot 12 = 36$

For $n=4$

$x(4) = 3x(\cancel{4}-1)$

$= 3x(3)$

$= 3 \times 36 = 108$

The General Form of eqn:

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

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

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

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

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

Given

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

$x(1) = 1$

$n = 2k$

$x(2k) = x(\frac{2k}{2})+2k$

$x(2k) = xk+2k$

Sub $k=1$

$x(2\cdot 1) = x(1) + 2(1)$

$= 1 + 2$

$x(2) = 3$

Sub $k=2$

$x(2\cdot 2) = x(2) + 2(2)$

$x(4) = 3 + 4 = 7$

Sub $k=3$

$x(2\cdot 3) = x(3) + 2(3)$

$x(6) = x(1.5) + 3$

$\therefore$ The general eq for given equation

$$\boxed{x(2k) = x(k) + 2k}$$

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

Given

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

$n = 3k$

$x(3k) = x\left(\frac{\cancel{3}k}{\cancel{3}}\right) + 1$

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

Sub $k=1$

$x(3\cdot 1) = x(1) + 1$

$= 1 + 1$

$= 2$

Sub $k=2$

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

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

$\therefore$ The General equation for $\boxed{x(3k) = 1 + \log_3(k)}$

2) evaluate the following recurrences completely

(i) $T(n) = T(n/2) + 1$, where $n = 2k$ for all $k \geq 0$

Given $n = 2k$, i.e $k = \log n$

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

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

$T(2 \cdot k) = T(k/2) + 2$ (if k is even)

$T(2 \cdot k) = T(k - 1/2) + 2$ (if k is odd)

$T(2 \cdot k) = T(1 \cdot k)$

$\Rightarrow$ Reccurrences $\Rightarrow$ $\boxed{T(n) = \Theta(\log n)}$

ii) $T(n) = T(n/3) + T(2n/3) + cn$, where 'c' is a constant and 'n' is the input size.

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

$a = 2, \quad b = 3, \quad f(n) = c(n)$

Master's theorem

$f(n) = O(n^c)$ where $c < \log_b a$

$T(n) = \Theta(n \log_b a)$

$f(n) = \Theta(n \log_b a)$

Then

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

$f(n) = \Omega(n^c)$

where $c > \log_b a$, $aT\left(\frac{n}{b}\right) \leq k f(n)$

for $k < 1$

$T(n) = \Theta(f(n))$

find $\log_b a = \log_b a = \log_3 2$

$f(n) = cn = n \log_b^a$

Recurrence Relation $\Rightarrow$ $\boxed{T(n) = O(n)}$.

3) consider the following recursion algorithm

```
Miny (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 A[n-1]
```

a) what does algorithm compute?

$\Rightarrow$ This algorithm computes the minimum element in an array A of size using a recursive approach.

$\Rightarrow$ Recursive Case:

* If the array has more than one ele $(n>1)$ min element in subarray consisting of first $(n-1)$ elements.

* The result of this recursive call ("Temp") is then compared to the last element of the current array segment ("A[n-1]")

* The function returns the smaller of these two values

b) Setup a recurrence relation for the algorithms basic operation count and solve it.

$T(n)$ = No of basic operations

if $n=1$ then $T(1) = 0$

"$T(n) = T(n-1) + 1$" is the recurrence relation

$T(1) = 0$

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

$= T(1) + 1$

$= 0 + 1 = 1$

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

$= 1 + 1 = 2$

$T(4) = T(4-1) + 1$

$= T(3) + 1$

$= 2 + 1 = 3$

$T(n) = n-1$

$\therefore$ Time complexity $= O(n)$, where n = size of array

4) Analyze the order of growth

(i) $F(n) = 2n^2 + 5$ and $g(n) = 7n$. Use the $\Omega(g(n))$ notation.

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

$g(n) = 7n$

if $n=1 \Rightarrow f(n) = 2(1)^2 + 5$ $\quad$ $g(n) = 7(1)$

$= 7$ $\quad$ $= 7$

$n=2 \Rightarrow f(n) = 2(2)^2 + 5$ $\quad$ $g(n) = 7(2)$

$= 13$ $\quad$ $= 14$

$n=3$

$\Rightarrow f(n) = 2(3)^2+5$ $\qquad g(n) = 7(3)$

$= 23$ $\qquad = 21$

$n=4 \Rightarrow f(n) = 2(4)^2+5$ $\qquad g(n) = 7(4)$

$= 2(16)+5$ $\qquad = 28$

$= 37$

$F(n) \geq g(n) \cdot c$ condition satisfies at $n=1$ onwards so the $\Omega(7n)$ is the occurence rln.

$\therefore$ Time complexity is $\Omega(n)$