1. Solve the following Recurrence Relations :

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

⇒ using back substitution,

⇒ $x(n) = x(n-1) + 5 \rightarrow$ ①

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

subs $x(n-1)$ in ①,

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

⇒ $x(n) = x(n-2) + 10 \rightarrow$ ②

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

$x(n-2) = x(n-3) + 5$

subs $x(n-2)$ in ②,

$x(n) = (x(n-3) + 5) + 10$

⇒ $x(n) = x(n-3) + 15$

⇒ Hence the observed sequence is,

$$x(n) = x(n-k) + 5k.$$

⇒ $n - k = 1$

$n - 1 = k.$

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

$$= 0 + (5(n-1))$$

$$= 5n - 5$$

$\Rightarrow$ The Time complexity is $O(n)$.

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

$\Rightarrow$ using back substitution,

$\Rightarrow x(n) = 3x(n-1) \rightarrow$ ①

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

subs $x(n-1)$ in ①,

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

$\Rightarrow x(n) = 9x(n-2) \rightarrow$ ②

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

subs $x(n-2)$ in ②,

$x(n) = 3^2(3x(n-3))$

$\Rightarrow x(n) = 3^3 x(n-3)$

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

→ The sequence is

$x(n) = 3^k x(n-k)$

→ $n-k = 1$

$n-1 = k$

subs, we get

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

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

→ The Time complexity is $O(3^n)$.

(iii) $x(n) = x(n/2)+n$ for $n>1$ $x(1)=1$

→ using masters theorem,

$x(n) = ax(n/b)+f(n)$.

Here, $a=1, b=2$.

$\log_a^b = \log_1^2 = 0$.

$p=1, k=1$.

→ Case I: $\log_a^b > k$,

→ $n^{\log_a^b}$

→ case 2b: $\log_a^b = K$

$P > -1 \rightarrow O(n^K \log^{P+1} n)$

$P = -1 \rightarrow O(n^K \log\log n)$

$P \leq -1 \rightarrow O(n^K)$.

→ case 3: $\log_a^b < K$.

$P \geq 0 \rightarrow O(n^K \log^P n)$

$P \leq 0 \rightarrow \Theta(n^K)$.

Here, case 3 is satisfied.

~~$P \leq 0 \rightarrow \Theta(n^K)$~~.

$P \geq 0 \rightarrow O(n^K \log^P n)$.

$1 \geq 0 \rightarrow O(n^1 \log^1 n)$.

→ The Time complexity is $O(n)$.

(iv) $x(n) = x(n/3) + 1$ for $n > 1$ $x(1) = 1$.

→ Tree method:

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

find $x(n/3) = x\left(n/3/3\right) + 1 = x\left(n/3^2\right) + 1$.

Applying log,

$$\Rightarrow \log_2 n = \log 3^{i}$$

$$\log_2 n = i \log_2 3$$

$$\Rightarrow \frac{\log_2 n}{\log_2 3} = i \quad , \quad \frac{\log 3^{n}}{\partial} = i$$

$$\Rightarrow x(n) = x\left(\frac{n}{3^{\log_3 n}}\right) + \log_3 n.$$

$n = 3k$

$$x(3k) = x\left(\frac{3k}{3^{\log_3 n}}\right) + \log_3 n.$$

$$= x(3k/n) + \log_3 n.$$

$$= x(3k/3k) + \log_3 n.$$

$$x(3k) = x(1) + \log_3 n.$$

$$\left.\begin{array}{l} x(3k) = \log_3 n+1 \\ x(n) = \log_3 n+1 \end{array}\right\} O(\log n).$$

$$\Rightarrow x(n) = (x(n/3^2)+1)+1$$

$$= (x(n/3^2)+2) \rightarrow (2)$$

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

subs in (2),

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

$$x(n) = x(n/3^3)+3 \rightarrow (3)$$

$$x(n/3^3) = x(n/3^3/3)+1 = x(n/3^4)+1.$$

subs in (3)

$$x(n) = (x(n/3^4)+1)+3 = x(n/3^4)+4.$$

→ The sequence is as follows:

$$x(n) = x\left(\frac{n}{3^i}\right)+i$$

$$\frac{n}{3^i} = 1. \qquad x(1) = 1.$$

$$n = 3^i$$

2. Evaluate the following recurrences completely:

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

⇒ Tree method.

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

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

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

⇒ $T(n) = T(n/2^2) + 1 + 1$

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

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

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

⇒ $T(n) = T(n/2^3) + 3$

$T(n/2^3) = T(n/2^4) + 1.$

⇒ $T(n) = T(n/2^4) + 4.$

observed series:

$T(n) = T(n/2^i) + i$

$\frac{n}{2^i} \Rightarrow n = 2^i$ where $i = \log_2 n$.

$T(n) = T(n/2^{\log_2 n}) + \log_2 n$.

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

$T(2k) = T\left(\frac{2k}{2k}\right) + \log_2 2k$.

$T(2k) = 1 + \log_2 k$

$T(2k) = 1 + \log_2 k$.

$\Rightarrow$ $O(\log K)$ is the time complexity.

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

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

$T(2n/3) = T(2n/3^2) + T(4n/3^2) + c\,2n/3^2$

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

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

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

$T(4n/3^2) = T(4n/3^3) + T(8n/3^2)$

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

$+ T\left(\frac{2n}{3^3}\right) + T(4n/3^3) + T\left(\frac{4n}{3^2}\right) + T\left(\frac{8n}{3^3}\right)$

3. consider the following recursion algorithm.

```
min 1 (A [0 --- n-1])
if n=1 return A [0]
Else temp = min 1 (A [0.... n-2])
    If temp ≤ A [n-1]
    return temp
Else
    return A(n-1)
```

a) what does this algorithm compute?

a) Base case [n = 1]

If the size of the array is 1, the algorithm returns the only element in the array.

b) Recursive [n-1]

If the size of the array is greater than 1, the algorithm recursively computes the minimum array of the subarray A [0, ... n-2].

c) comparison:

computing the minimum of the subarray and comparing the minimum value with the last element of the array A [n-1].

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

b) Recurrence Relation :

$c = n.$

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

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

$T(n) = T(n-1) \rightarrow 1$

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

$= T(n-2) + 2c \rightarrow 2$

sub 2 in 1,

$T(n) = T(n-2) \rightarrow 3$

Find $T(n-2) = T(n-2-1) \Rightarrow T(n-3) \rightarrow 4.$

sub 4 in 3,

$T(n) = T(n-3) + 3c.$

* Series : $T(n-2), T(n-3).$

$T(1) = 1.$

$T(n) = T(n-k) + kc.$

$n-k = 0$	$n-k = 1$
$n = k.$	$n-1 = k.$

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

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

⇒ The Time complexity is $O(n)$.

4. Analyse the order of growth

i) $f(n) = 2n \cdot 2+5$ and $g(n) = 7n$. use the $\Omega(g(n))$ notation.

The Equation for $\Omega(g(n))$ is $f(n) \geq g(n).c$.

$$f(n) = 2n^2+5 \text{ and } g(n) = 7n.$$

$$2n^2+5 \geq 7n.$$

$n=1$, $2+5 \geq 7n \Rightarrow 7=7$

$n=2$, $8+5 \geq 14 \Rightarrow 13 \leq 14$

$n=3$, $2(9)+5 = 7\times3 \Rightarrow 23 \geq 21$.

⇒ condition satisfied.

⇒ $f(n)$ is growing faster than $g(n)$.

$\Omega(7n)$.