1. a)

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

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

by substitutional method

$x(1) = 0$ — (1)

If $n = 2 \Rightarrow x(2) = x(2-1) + 5 = x(1) + 5$ — (2)

$x(1) = 0 + 5$

$x(1) = 5$ — (3)

If $n = 3 \Rightarrow x(3) = x(3-1) + 5$

$= x(2) + 5$ — 4

Substitute (3) in (4)

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

$x(n) = 5n$ for $n > 1$

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

$x(1) = 4$ — (1)

If $n = 2 \Rightarrow x(2) = 3x(2-1) = 3x(1)$ — (2)

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

Substitute (1) in (2)

$x(2) = 3(4) = 12$ — (3)

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

$x(3) = 3x(2)$ — 4

Substitute (3) in (4)

$x(3) = 3 \times 12 = 36$

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

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

$x(n) = x(n/2) + n$ i.e $= n = 2^k$

$x(1) = 1$

Substitute $n = 2^k$

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

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

$x(2^0) = 1$

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

$= x(2^0) + 2^1 = 1 + 2 = 3$

$x(2^1) = 3$

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

$= x(2^1) + 4 = 3 + 4 = 7$

$x(2^2) = 7$

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

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

$x(n) = x(n/3) + 1$ — (1)

$x(1) = 1$ $n = 3^k$

Substitute $n = 3^k$ in (1)

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

$= x(3^{k-1}) + 1$

$x(3^0) = x(1) = 1$

$x(3^1) = x(3^{1-1}) + 1 = 1 + 1 = 2$

$x(3^1) = 3$

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

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

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

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

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

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

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

$T(n)$

$T(n/3$ $\quad$ $T(2n/3)$

$T(n/9)$ $\quad$ $T(2n/9$ $\quad$ $T(2n/9)$ $\quad$ $T(4n/9)$

$T(1)$ $\quad$ $T(1)$ $\quad$ $T(1)$ $\quad$ $T(1)$

cn

$c(n/3)$ $\quad$ $c\,2n/3)$

$c(n/9)$ $\quad$ $c\,n/9)$ $\quad$ $c(2n/9)$ $\quad$ $c\,4n/9)$

c $\quad$ c $\quad$ c $\quad$ c

length $= \log_3 4$ (divide by 3)

$$T(n) = cn\log_3 4$$

$$= w(n\log n)$$

3: a) what does this algorithm compute?

The algorithm compute minimum elements in arr of A of sign,

If $i < n$, $A[i]$ is smaller than all elements then

$A[j]$s $j = i+1$ to $n-1$, then it return

$A[j]$,

It also return the left most minimal element

(b) Setup a recurrence relation for the algorithm basic operation count and solve it

mainly comparision occurs during recursion

so, $T(n) = T(n-1)+1$, where $n>1$

$T(1) = 0$ (when $n=1$

no comparision)

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

$= 0 + (n-1$

$T(n) = n-1$

time complexity $= O(n)$

7. (i) $f(n) = 2n^2+5$ and $g(n) = 7n$

use the Ω $(g(n))$ notation

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

$c.g(n) = 7n$

$f(n) \geq c.g(n)$

$n = 1$

$f(1) = 2(1)^2+5$

$= 2+5$

$= 7$

$c.g(n) = 7n$

$= 7(1)$

$= 7$

$n = 2$

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

$= 8+5 = 13$

$c.g(n) = 7n$

$= 7\times2$

$= 14$

$n = 3$

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

$= 18+5 = 23$

$c.g(n) = 7(n)$

$= 7\times3 = 21$

$n = 1,\ 7 = 7$

$n = 2,\ 13 = 14$

$n = 3\ 23 = 1$

$n \geq 3,\ f(n) \geq c.g(n)$

f(n) is always greater than or equal to c.g(n) when n value is greater or equal to 3

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

f(n) is grows more than g(n) from below asymptotically