

Name: Ganesh Suresh  
Section: F  
Roll No: 57

## Tutorial - 2

①

Ans 1. `void func (int n)`  
{  
  int j = 1, i = 0;  
  while (i < n)  
  {  
    i = i + j;  
    j++;  
  }  
}

$$\begin{aligned} j=1, & i=0+1 \\ j=2, & i=0+1+2 \\ j=3, & i=0+1+2+3 \end{aligned}$$

loop ends when  $i \geq n$   
 $0+1+2+3+\dots+n > n$   
 $\frac{k(k+1)}{2} > n$

$$\begin{aligned} k^2 &> n \\ k &> \sqrt{n} \end{aligned}$$

~~$O(n)$~~   $O(\sqrt{n})$

Ans 2. Recurrence Relation F. & S Fibonacci Series

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

$$T(0) = T(1) = 1$$

if  $T(n-1) \approx T(n-2)$

(Lower  
Bound)

$$T(n) = 2T(n-2)$$

$$= 2 \{ 2T(n-4) \} = 4T(n-4)$$

$$= 4(2T(n-6))$$

$$= 8T(n-8)$$

$$= 8(2T(n-8))$$

$$= 16T(n-8)$$

$$T(n) = 2^K T(n-2K)$$

$$n-2K = 0$$

$$n = 2K$$

$$K = \frac{n}{2}$$

$$\begin{aligned} T(n) &= 2^{n/2} T(0) \\ &= 2^{n/2} \end{aligned}$$

$$T(n) = \Omega(2^{n/2})$$

②

• If  $T(n-2) \approx T(n-1)$

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

$$= 2(2T(n-2)) = 4T(n-2)$$

$$= 4(2T(n-3)) = 8T(n-3)$$

$$= 2^K T(n-K)$$

$$n-K = 0$$

$$K = n$$

$$T(n) = 2^K \times T(0) = 2^n$$

$$= T(n) = O(2^n) \quad (\text{upper bound})$$

Ans 3 •  $O(n(\log n)) \Rightarrow$  for (int i = 0; i < n; i++)  
 {  
   for (int i = 1; j < n; j = j \* 2)  
   {  
     // some  $O(1)$   
   }  
}

•  $O(n^3) \Rightarrow$  for (int i = 0; i < n; i++)  
 {  
   for (int j = 0; j < n; j++)  
   {  
     for (int ~~j~~<sup>k</sup> = 0; k < n; k++)  
     {  
       // some  $O(1)$   
     }  
   }  
}

Ans 4 ~~int~~ •  $T(n) = T(n/4) + T(n/2) + cn^2$

• Let's assume  $T(n/2) \geq T(n/4)$

$$\text{So, } T(n) = 2T(n/2) + cn^2$$

applying master's Theorem ( $T(n) = aT(\frac{n}{b}) + f(n)$ )

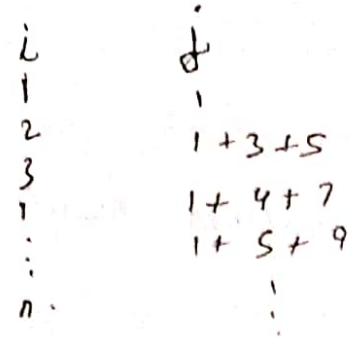
$$a = 2, b = 2, f(n) = n^2$$

$$c = \log b^a = \log 2^2 = 1$$

compare  $n^c$  and  $n^{2-n}$   
 $f(n) > n^2$  so,  $T(n) = O(n^2)$

$$f(n) > n^2 \text{ so, } T(n) = O(n^2)$$

Ans 5. for



$$j = (n-1) / i \text{ times}$$

$$\sum_{i=1}^n \frac{(n-1)}{i}$$

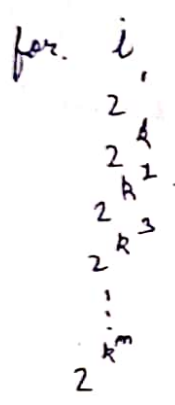
$$\therefore T(n) = \frac{(n-1)}{1} + \frac{(n-1)}{2} + \frac{(n-2)}{3} + \dots + \frac{(n-1)}{n}$$

$$T(n) = n \left[ 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \right] - 1 \times \left[ 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \right]$$

$$= n \log n - \log n$$

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

Ans 6 :-



where

$$2^{k^m} \leq n$$

$$k^m = \log_2 n.$$

$$m = \log k \log_2 n.$$

$$\therefore \sum_{i=1}^m 1$$

$$1+1+1 \dots m \text{ times}$$

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

Ans 7. Given algorithm divides array in 99% and 1% part

$$\therefore T(n) = T(n-1) + O(1)$$

$$T(n) = (T(n-1) + T(n-2) + \dots + T(1) + O(1)) \times n$$

$$= n \times n$$

$$\therefore T(n) = O(n^2)$$

lowest height = 2

highest height = n.

(4)

$\therefore$  difference =  $n-2$   $n > 2$

The given algorithm produces linear result

Ans 8. (a)  $n, n!, \log n, \log \log n, \text{root}(n), \log(n!), n \log n, \log^2(n), 2^n, 2^{2^n}, 4^n, n^2, 100$

$$100 < \log \log n < \log n < (\log n)^2 < \sqrt{n} < n < n \log n < \log(n!) < n^2 < 2^n < 4^n < 2^{2^n}$$

(b)  $2(2^n), 4n, 2n, 1, \log(n), \log(\log(n)), \sqrt{\log(n)}, \log 2n, 2\log(n), n, \log(n!), n!, n^2, n \log(n)$ .

$$1 < \log \log n < \sqrt{\log n} < \log n < \log 2n < 2\log n < n < n \log n < 2n < 4n < \log(n!) < n^2 < n! < 2^{2^n}$$

(c)  $8^{2^n}, \log_2(n), n \log_6(n), n \log_2(n), \log(n!), n!, \log_2(n), 96, 8n^2, 7n^3, 5n$ .

$$96 < \log_3 n < \log 2n < 5n < n \log_6(n) < n \log_2 n < \log(n!) < 8n^2 < 7n^3 < n! < 8^{2^n}$$