TUTORIAL-02

Answer-1 :-
```
void fun(int n){
    int j=1, i=0;
    while (i<n){
        i=i+j;
        j++;
    }
}
```

i	j
0	1
1	2
3	3
6	4
10	5
15	6

Series: 0, 1, 3, 6, 10, 15 - - - -

$n = 0+1+2+3+\cdots+k$

$n = \frac{k(k+1)}{2}$

$n = \frac{k^2+k}{2}$

$n \cong k^2$

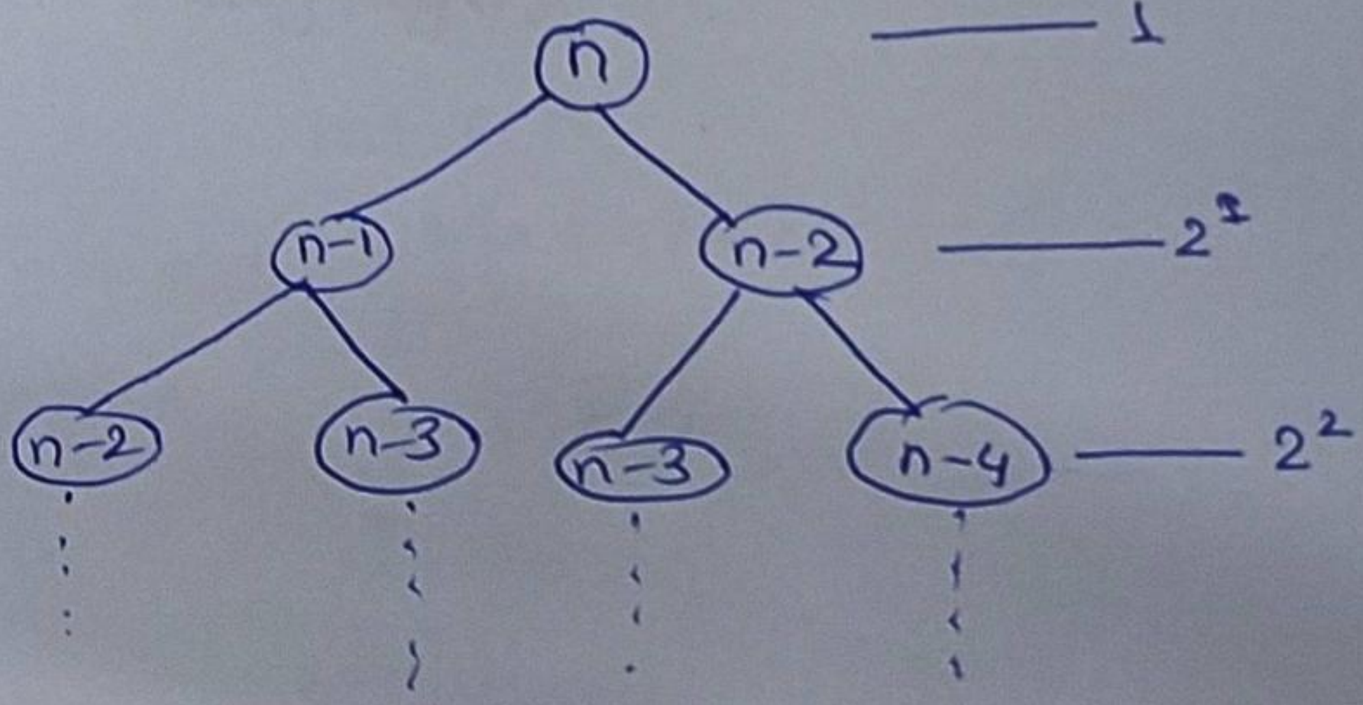
$k \cong \sqrt{n}$

Time complexity = $O(\sqrt{n})$

nswer-2. Recurrence relation for fibonacci series

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

Using Recurrence tree method



Time complexity = $1 + 2 + 4 + \cdots + 2^n$

$= \frac{1(2^{n+1}-1)}{2-1} = 2^{n+1}-1$

or, Time complexity = $O(2^n)$

Space complexity – Space complexity of fibonacci series using recursion is proportional to hight of recurrence tree.

or, Space complexity = $O(n)$

Answer-3:- write code for complexity

(i) nlogn

```
For(i to n)
  {  for (j=1; j<=n; j*=2)
          O(1) // Statements
  }
```

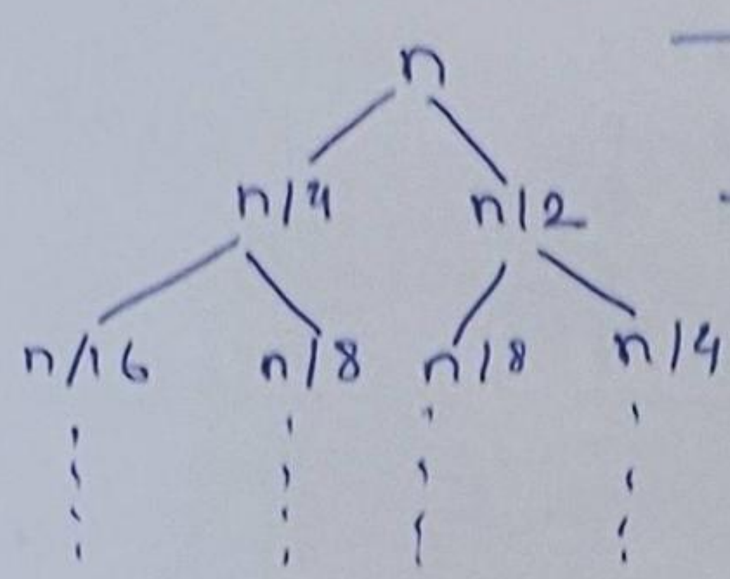
(ii) n^3

```
For(i to n)
    for (j to n)
        for (k to n)
            O(1)  // Statements
```

(iii) log(log n)

```
int i=n;
while (i>0)
 {
   j=√i;
 }
```

Answer-4:- $T(n) = T(n/4) + T(n/2) + \cdots cn^2$



cn^2

$\frac{cn^2}{16} + \frac{cn^2}{4} = \frac{5cn^2}{16}$

$\frac{cn^2}{256} + \frac{cn^2}{64} + \frac{cn^2}{64} + \frac{cn^2}{16} = \frac{25}{256} cn^2$

So, $T(n) = c\left(n^2 + \frac{5n^2}{16} + \frac{25n^2}{256} + \cdots\right)$

here, $r = \frac{5}{16}$, so, $S_n = \frac{1}{1-r}$

$$T(n) = cn^2\left(1 + \frac{5}{16} + \frac{25}{256} + \cdots\right)$$

$$= cn^2\left(\frac{1}{1-5/16}\right)$$

$$= cn^2 \times \frac{16}{11}$$

Time complexity $= \theta(n^2)$

Answer-5!-
```
int fun (int n) {
    For (int i = 1 ; i = n; i++) {
        For (int j = 1 ; j <= n; j += i) {
            // some O(1) task
        }
    }
}
```

i	j	time
1	1 to n	n-1
2	1 to n	(n-1)/2
3	1 to n	(n-1)/3
!	!	
n	1 → n	(n-1)/n

nlogn

Time complexity = O(nlogn)

Answer-6:-

```
for(int i= 2; i<=n; i=pow(i,k))
    {
      // some  O(1) expressions or statements
    }
```

$i = 2, 2^k, 2^{k^2}, 2^{k^3} \dots 2^{k^x}$

$n = 2^{k^x}$

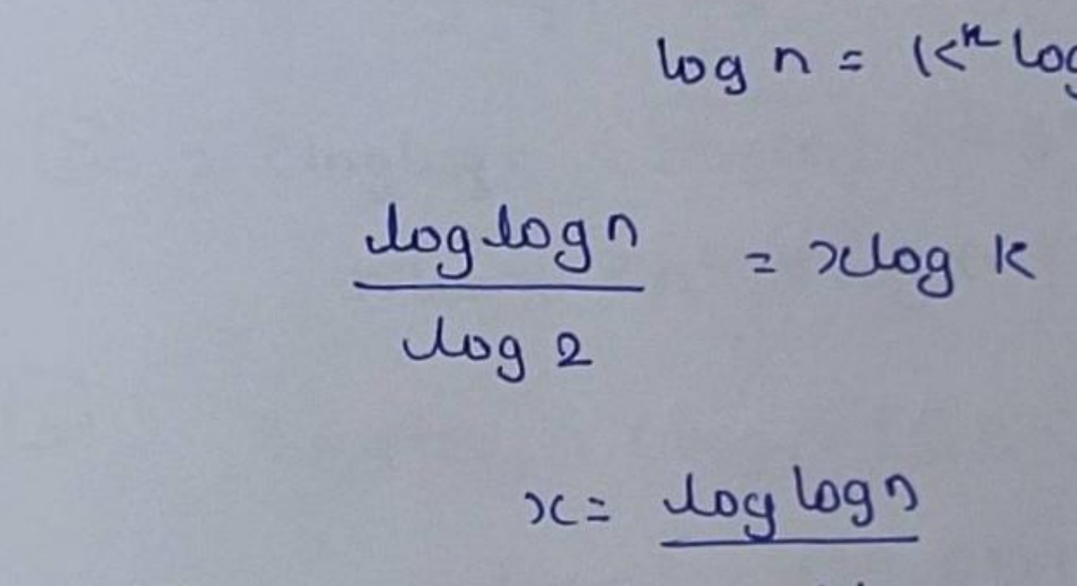
$\log n = k^x \log 2$

$\frac{\log \log n}{\log 2} = x \log k$

$x = \frac{\log \log n}{\log 2 + \log k}$

Time Complexity = O(loglogn)

Answer-7:-



Taking longer branch that is $\frac{99n}{100}$

Time complexity $= \log_{\frac{100}{99}} n$

$\simeq \log n$

$n = \left(\frac{99}{100}\right)^k$

or $k = \log\left(\frac{100}{99} n\right)$

$T(n) = n\left(\log_{\frac{100}{99}}\right)^n / 100$

$= O(n \log_{99} n)$

8) Increasing order of rate of growth

(a) $n, n!, \log n, \log\log n, \text{root}(n), \log(n!), n\log n, \log^2 n,$

$\log \cdot 2^n, 2^{2^n}, 4^n, n^2, 100$

$100 < \log\log n < \log n < \sqrt{n}\,(\text{root}(n)) < n < n\log n < n^2 < 2^n < 2^{2^n} < 4^n(n?$

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

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

(c) $96 < \log_8 n < \log_2 n < 5n < n\log_8 n! < n\log_2 n < 8n^2 < 7n^3 <$

$\log n! < 8^{2n} < n!$