Toutorial Sheet - 02

Sol1

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

series = 0, 1, 3, 6, 10, 15 . . . .

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

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

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

$n \simeq k^2$

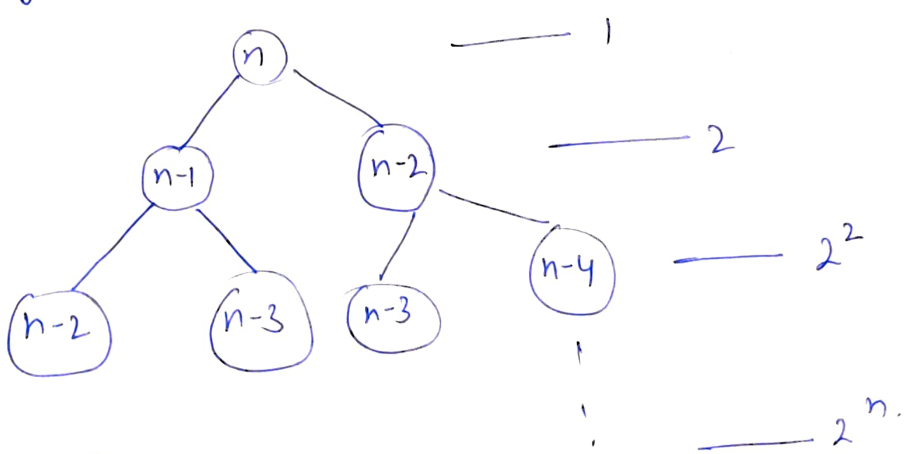
$k \simeq \sqrt{n}$

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

Ques2 Recurrence Relation for fibonacci series.

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

using recursion tree method



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

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

or $TC = O(2^n)$

Space Complexity: space complexity of fibonacii series using recursion is proportional to height of recurrence tree

Space complexity = $O(n)$

Ques3 Write code for complexity :-

(i) $n \log n$.

```
for(i=1; i<=n; ++i)
 {
   for (j=1; j<=n; j*=2)
    {
      O(1) statement
    }
 }
```

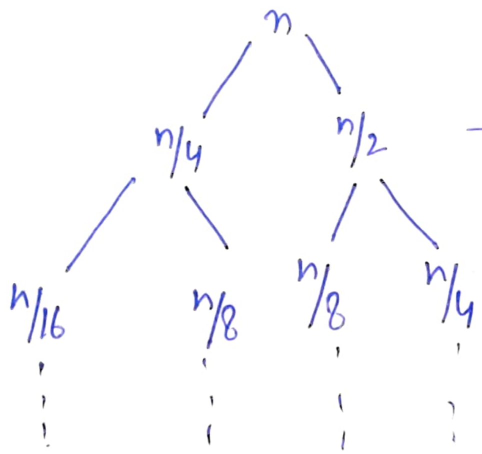
(ii) n^3

```
for (i=1; i<=n ; ++i)
 {
   for(j=1; j<=n; ++j)
    {
      for(k=1; k<=n; ++k)
       {
         O(1) statement;
       }
    }
 }
```

(iii) $\log(\log n)$

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

Q4 $T(n) = T(n/4) + T(n/2) + Cn^2$



— Cn^2

— $\frac{Cn^2}{16} + C\frac{n^2}{4} = \frac{5Cn^2}{16}$

— $\frac{Cn^2}{256} + \frac{Cn^2}{64} + \frac{Cn^2}{64} + \frac{Cn^2}{16}$

$= \frac{25}{256}Cn^2$

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

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

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

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

$= Cn^2 \times \frac{16}{11} = n^2$

$\underline{TC = O(n^2)}$

Q5

```
int func (intn){
    for(int i=1; i<=n; ++i){
        for (int j =1 ; j<=n; j+=1){
            .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 to n	(n-1)/n

$$\frac{(n-1)/n}{n \log n}$$

Time complexity = $O(n \log n)$

Ques for (int i=2; i<=n; i= pow (i,k))
{
// some O(1) expression.
}.

$i = 2, 2^k, 2^{k^2}, 2^{k^3} \ldots 2^{k^n}$

$n = 2^{k^n}$

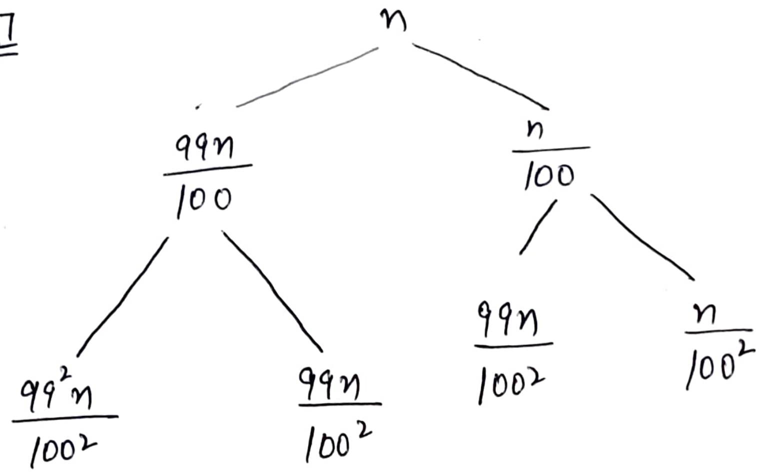
$\log n = k^n \log 2$

$\frac{\log . \log n}{\log 2} = n \log n$

$n = \frac{\log \log n}{\log 2 * \log n}$

$\boxed{TC = O(\log \log n)}$

Q7



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

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

$\cong \log n$

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

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

$$T(n) = n\left(\log \frac{900}{99}\right)^n / 100$$

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

Ques2 Increasing order of rate of growth

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

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

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