Tutorial Sheet - 2

Q1)

```
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
	7

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

$$n = 0+1+2+3+ \dots +K$$

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

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

$$n \cong K^2$$

$$K \cong \sqrt{n}$$

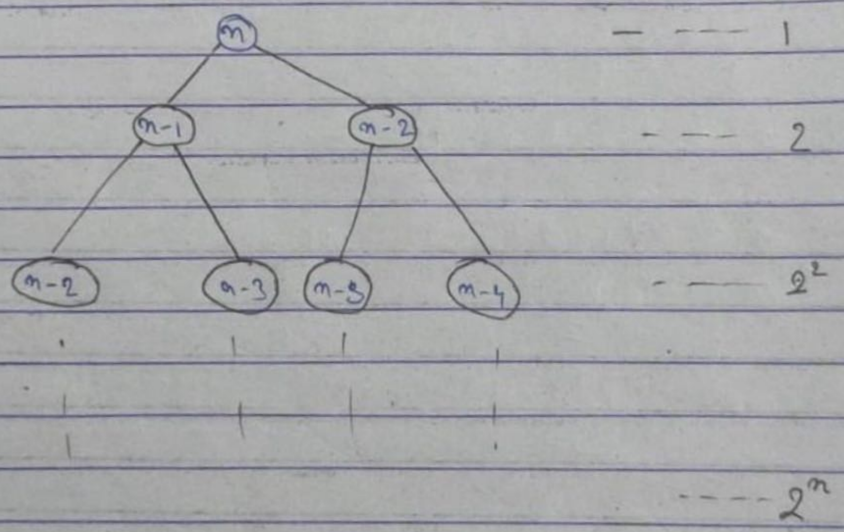
Time Complexity = $O(\sqrt{n})$

Q2>

Recurrence relation for fibonacci series

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

Using Recurrence tree method



Time Complexity $= 1 + 2 + 4 + \dots + 2^n$

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

or Time Complexity $= O(2^n)$

Space Complexity : Space Complexity of fibonacci series using recursion is proportional to height of recurrence tree.

So, Space Complexity = $O(n)$

Q3) Write code for Complexity

(i) $n \log n$

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

(ii) n^3

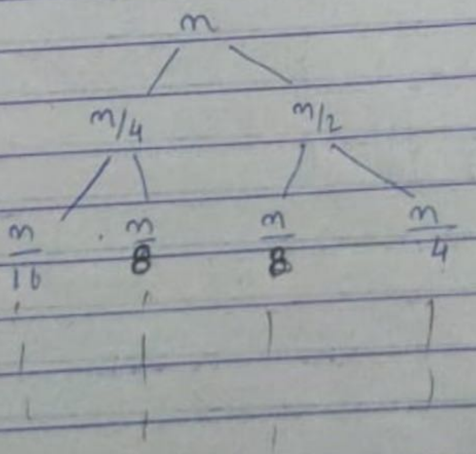
```
for (i to n)
    for (j to n)
        for (k to n)
            O(1) statements
```

9:03 PM

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

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

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



$- \quad cn^2$

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

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

$\frac{25}{256} cn^2$

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

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

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

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

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

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

Q.5)

```
int fun (int n) {
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j < n; j += i) {
            //Some O(1) task
        }
    }
}
```

9:03 PM

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
		n log n

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

Q6>

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

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

$n = 2^{k^x}$

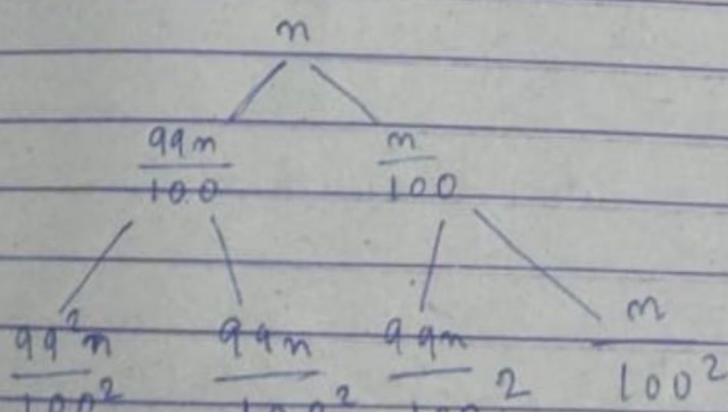
$\log n = k^x \log 2$

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

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

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

S7)



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

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

$\cong \log n$

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

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

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

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

Q8) 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

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

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

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