Tutorial - 2

Ans1)

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

$j = 1$, $i = 0 + 1$

$j = 2$, $i = 0 + 1 + 2$

$j = 3$; $i = 0 + 1 + 2 + 3$

⋮

loop ends when $i >= n$

$0 + 1 + 2 + \dots n \geq n$

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

$k^2 > n$

$O(\sqrt{n})$

Ans2) Reccurance 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) $T(n) = 2T(n-2)$

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

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

$= 8T(n-6)$

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

$= 16T(n-8)$

⋮

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

$n - 2k = 0$

$n = 2k$

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

$T(n) = 2^{n/2} T(0)$

$= 2^{n/2}$

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

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

$$T(n) = 2T(n-1)$$
$$= 2(2T(n-2)) = 4T(n-2)$$
$$= 4(2T(n-3)) = 8T(n-3)$$
$$= 2^k T(n-k)$$

$$n-k=0$$
$$\boxed{k=n}$$

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

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

Ans3) • $O(n\log n) \Rightarrow$

```
for (int i=0; i<n; i++)
{
    for (int j=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 k=0; k<n; k++)
        {
            // Some O(1)
        }
    }
}
```

• $O(\log(\log n)) \Rightarrow$

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

Ans4) $T(n) = T(\frac{n}{4}) + T(\frac{n}{2}) + cn^2$

- lets assume $T(n/2) >= T(n/4)$

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

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

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

$c = \log_b a = \log_2 2 = 1$

$n^c = n$

compare n^c and $f(n) = n^2$

$f(n) > n^c$

So, $T(n) = \Theta(n^2)$

Ans5)
```
int fun(int n)
{
  for(int i=1; i<=n; i++)
  {
    for(int j=1; j<n ;j+=1)
    {
       // some O(1)
    }
  }
}
```

$i = 1$ — $j=1$, $j=2$, $j=3$ ... $j \ge n$ — n times

$i = 2$ — $j=1$, $j=3$, $j=5$, $j=7$ — n times

Loop ends when $j \ge n$

$1+3+5+7 > n$

$k > \frac{n}{2}$

$i = 3$ — $j=1$, $j=4$, $j=7$ — $1+4+7 > n$

$k > \frac{n}{4}$

⋮

$i = n$

So, Total complexities $= O(n^2 + n^2 + n^2 + \dots)$
$= O(n^2)$

Ans6) for(int i=2; i<=n; i = Pow(i, k))
{
$1 sam(1)
}

complexity of Pow(i,k) — O(log N)
= log(K)

$i = 2$
$i = 2^k$
$i = 2^{k^2}$
$i = \cancel{2k}\ 2^{k^3}$
$i = 2^{k^4}$
$\vdots$
$i = 2^{k^m}$

Loop ends when $i > n$
$\cancel{2k}\ 2^{k^M} > n$
$\log(2^{k^M}) > \log n$
$k^M \log 2 > \log n$
$k^M > \log n$
$\log k^M > \log(\log n)$
$M \log k > \log(\log n)$
$M > \dfrac{\log(\log n)}{\log(k)}$

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

Ans-8)

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

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

(c) $96 < \log_8 N < \log_n N < n\log_6 N < n\log_2 N < \log n! < N! < 5N <$
$8N^2 < 7N^3 < 8^{2n}$