Tutorial - 2

1

(a)

i	j
1	2
3	3
6	4
10	5

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

$$k^2 = n$$

$$k = \sqrt{n}$$

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

(b)

$T(0) = 0$

$T(1) = 0$

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

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

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

backward Substitution

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

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

$T(n-2) = 2T(n-3) + 1$

$= 2(2(2(T(n-3)+1)+1)+1)$

$= 8T(n-3) + 7$

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

$T(0) = 0$

$n - k \neq 0$

$n = k$

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

$= 2^n + 2^n$

$= O(2^n)$

(C) n^2

```
void fun(int n)
{
  for(int i=1; i<=n; i++)
  {
    for(int j=1, j<=n; j++)
    {  O(1) tasks
    }
  }
}
```

→ n^3

```
void func(int n)
{
  for(int i=1 ; i<=n; i++)
  {
  for(int j=1; j<=n; j++)
  {
    for(int k=1; j<=n; k++)
    {
        O(1) tasks
    }
  }
}
```

→ $\log(\log n)$

```
void func(int n)
{
   for(int i=n; i>1 ; i=pow(i,k))
   {   O(1) tasks
   }
}
```

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

assume $T\left(\frac{n}{2}\right) >= T\left(\frac{n}{4}\right)$

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

$c = \log_b a$

$c = \log_2 2 = 1$

$n^c < f(n)$

Time complexity $= O(n^2)$

(e)

i	j
1	n times
2	n/2 times
3	n/3 times
⋮	⋮
n	n/n times

→ T.C $= O(n \log n)$

(f) $i = 2, 2^k, (2^k)^k, (2^{k^2})^k = 2^{k^3} \quad \ldots \quad 2^{k \log_k (\log n)}$

$2^{k \log_k (\log n)} = n$

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