Name : Palak Tayal
Section : F
Roll no : 18

Design and Analysis of Algorithm
Tutorial - 2

Que1

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

j = 1, 2, 3, 4, ..., k
i = 1, 3, 6, 10, ... k

$1+2+3+....+k < n$

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

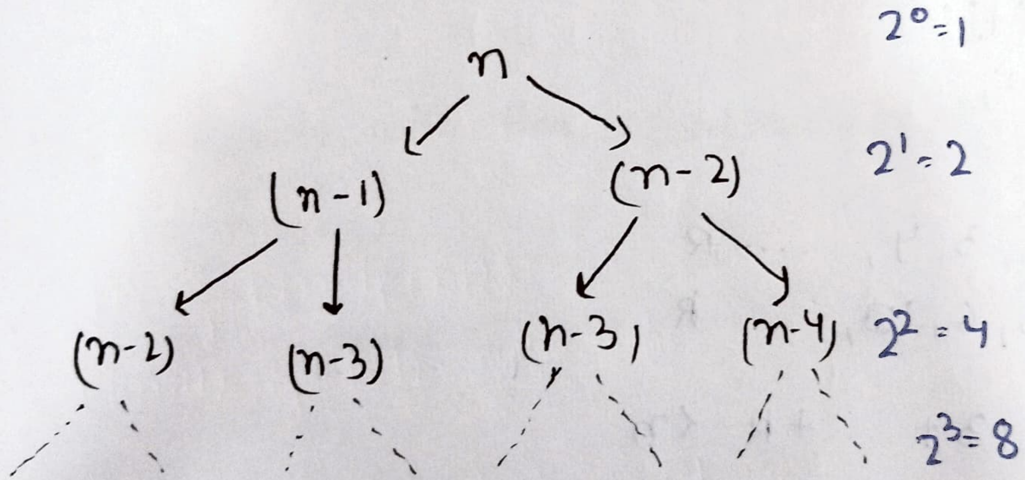
$k^2 < n$

$k = \sqrt{n}$

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

Que2 Fibonacci series with recursion

```
int fibbo (int n)
{   if (n==1)
        return 1;
    else if (n==0)
        return 0;
        else
            return fibbo(n-1) + fibbo(n-2);
}
```



Time Complexity = $O(2^n)$

Space Complexity = $O(1)$

using Recursion f^n

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

for $n=0$, $T(0)=1$ > only one $cond^n$ no recursive calls

for $n=1$, $T(1)=1$

Que3 (i) Program with Time complexity [n.log(n)]

```
int main()
{
    int n, count=0;
    cin>>n;
    for (int i=0; i<n; i++)
    {
        for(int j=0; j<n; j*=2)
            count++;
        cout << count <<endl;
    }
```

(ii) Program with time Complexity (n^3)

```
int main()
{
    int n, count=0;
    cin>>n;
    for (int i=0; i<n; i++)
    {
        for(int j=0; j<n; j+=2)
        {
            for (int k=0; k<n; k++)
                count++;
        }
    }
    cout <<count<<endl;
}
```

(iii) Program with time complexity log (log n)

```
int main()
{
    int n, p=0;
    cin>>n;
    for(int i=0; i<n; i*=2)
        p++;
    for(int j=1; j<P; j*=2
        cout<<j;
}
```

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

T(n/4) will be ignored as it is of lower order

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

put $n = \frac{n}{2}$ in eq^n (1)

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

put value of T(n/2) in eq^n (1)

$T(n) = T\left(\frac{n}{2^2}\right) + C\frac{n^2}{2^2} + Cn^2$ --- (II)

put $n = \frac{n}{4}$ in eq^n (1)

$T\left(\frac{n}{4}\right) = T\left(\frac{n}{2^3}\right) + C\frac{n^2}{2^4}$

Put value of $T\left(\frac{n}{4}\right)$ in eq^n (II)

$T(n) = T\left(\frac{n}{2^3}\right) + \frac{Cn^2}{2^4} + \frac{Cn^2}{2^2} + \frac{Cn^2}{2^0}$ ---- (III)

$T(n) = T\left(\frac{n}{2^k}\right) + Cn^2\left[\frac{1}{4^k} + \frac{1}{4^{k-1}} + \cdots + \frac{1}{4} + 1\right]$

$\frac{n}{2^k} = 1$

$2^k = n$

$R = \log n$

$$= T\left(\frac{n}{n}\right) + Cn^2\left[1+\frac{1}{4}+\frac{1}{16}+\cdots\frac{1}{4^k}\right]$$

$$= 1 + Cn^2\left[\frac{1-1/4^k}{1-1/4}\right]$$

$$= 1 + Cn^2 \times \frac{4}{3}\left[1 - \frac{4}{3} \times \frac{1}{2^{2k}}\right]$$

$$= 1 + Cn^2 \times \frac{4}{3}\left[1 - \frac{4}{3.n^2}\right]$$

$$= 1 + \frac{4}{3}Cn^2 - Cn^2 \times \frac{4}{3} \times \frac{4}{3n^2}$$

$$= 1 + Cn^2 \times \frac{4}{3} - C \times \frac{4}{3}$$

$$\boxed{T(n) = O(n^2)}$$

Que5

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

i	j	
1	1, 2, 3, . . . ,	n times
2	1, 3, 5, ... n	n/2 times
3	1, 4, 7 . . . . . n	n/3 times
⋮		
n	1. . . . n/n	1 time

Total time complexity ⇒ $n + \frac{n}{2} + \frac{n}{3} + \dots + \frac{n}{n}$

⇒ $n\left[1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}\right]$

$= n \sum_{k=1}^{n} 1/k$

$= n.\log n$

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

Ques6 for(int i=2; i<n; i= pow(i,k))

```
{
  //O(1)
}
```

$i \rightarrow 2^{k}, 2^{2k}, 2^{3k} \dots\dots 2^{ki}$

for termination of loop

$2^{ki} = n$

taking log,

$k^{i} \log_2 2 = \log_2 n$

$k^{i} = \log n$

taking log,

$i \log_k k = \log_k \log_2 n$

$i = \log_k \log_2 n$

complexity = $O(\log_k \log_2 n)$