# Tutorial - 2 (DAA)

Q1

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

Sol for $j = 1$ $\quad i = 1$

$j = 2$ $\quad i = 1 + 2$

$j = 3$ $\quad i = 1 + 2 + 3$ ] m levels

for (i)

$1 + 2 + 3 + \cdots \propto n$

$1 + 2 + 3 + m \propto n$

$\frac{m(m+1)}{2} \propto n$

$m \simeq \sqrt{n}$

∴ By summation

$\Rightarrow \sum_{i=1}^{m} 1 \Rightarrow 1 + 1 + 1 \cdots \sqrt{n}$ times

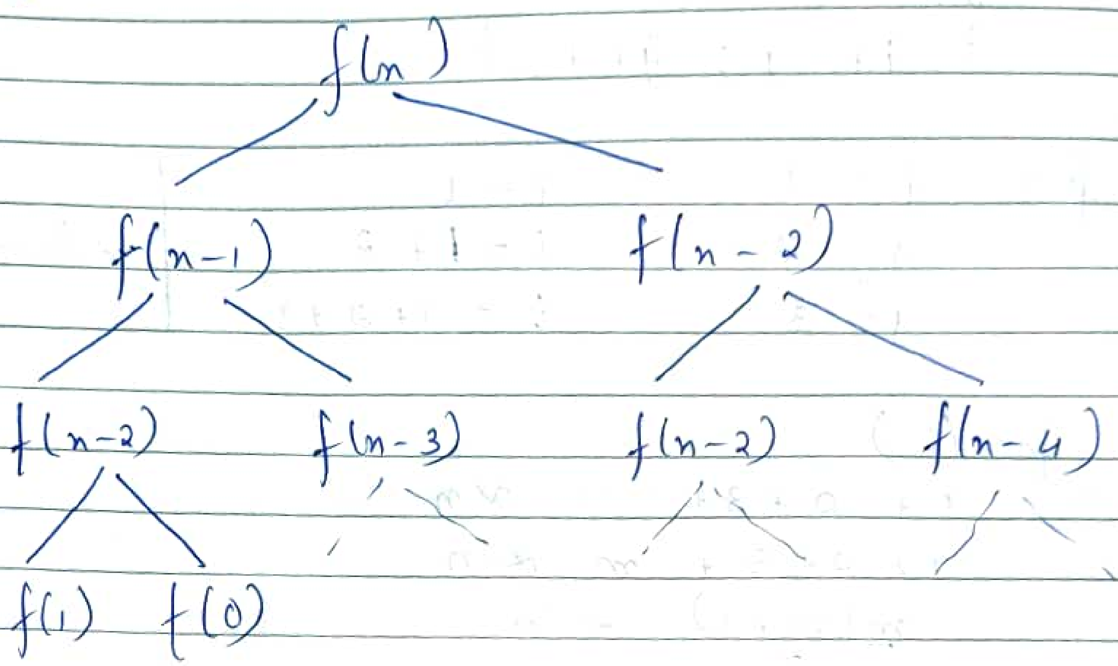
$$\boxed{T(n) = \sqrt{n}}$$

Q2) For Fibonacci series

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

$f(0) = 0$
$f(1) = 1$

Sol By forming tree




$\therefore$ At every func. call we got 2 func. call

for n levels

$= 2 \times 2 \times \ldots n$ times

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

Max Space:
Considering recursive stack
No. of calls max $= n$.
For each call
Space Complexity $= O(1)$
$T(n) = O(n)$
Without considering recursive stack:
Space Complexity $= O(1)$
$T(n) = O(1)$

classmate

Date
Page

Q3

① n log n, n^3, log(log n)

Sol Quick Sort

```
void QuickSort (int arr[ ], int low, int high)
 {
     if (low < high)
     {
         int pi = partition (arr, low, high);
         QuickSort (arr, low, pi-1);
         Quick Sort (arr, pi+1, high);
     }
 }
int partition( int arr[ ], int low, int high)
 {
     int part = arr[high];
     int i = (low - 1);

     for (int j = low ; j < = high - 1 ; j++)
     {
         if (arr[i] < part)
         {
             i++;
             swap (&arr[i], & arr[j]);
         }
     }
     swap (& arr [i+1], & arr [high]);
     return (i+1);
 }
```

(2) $n^3$

Multiplication of 2 square matrix

```
for(i=0; i<r1 ; i++)
    for(j=0; j<c2 ; j++)
        for (k=0; k<c1 ; k++)
        {
            res[i][j]+ = a[i][k] * b[k][j]
        }
```

(3) log(log n)

```
for (i=2; i<n ; i=i*i)
{
    count++;
}
```

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

Ans ⇒

$cn^2$

$T(n/4)$ $T(n/2)$

Further breaking $T(n/4)$ & $T(n/2)$

$cn^2$

$c(n^2)/16$ $c(n^2)/4$

$T(n/16)$ $T(n/8)$ $T(n/8)$ $T(n/4)$

Further ⇒ $c(n^2)/256$ $c(n^2)/64$ $c(n^2)/64$ $c(n^2)/16$

Summation level by level.

$T(n) = c(n^2 + 5(n^2)/16 + 25(n^2)/256) +$

G.P. ⇒ $r = \frac{5}{16}$

To get upper bound we can sum above series for infinite term.

$$Sum = \frac{(n^2)}{(1 - 5/16)}$$

$T(n) ⇒ O(n^2)$

Q5

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

Sol

| For i = 1 | Inner Loop (j) |
|---|---|
| 1 | $n$ |
| 2 | $n/2$ |
| 3 | $n/3$ |
| 4 | $n/4$ |
| ⋮ | |
| $n$ | $n/n$ |

Total Time Complexity $= \left(n + \frac{n}{2} + \frac{n}{3} + \dots + \frac{n}{n}\right)$

$$= n * \left(1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}\right)$$

$$\cancel{T(n) =} \; O(\log n)$$

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

classmate
Date
Page

Q6 Time Complexity:

```
for (int i = 2; i <= n; i = pow(i, k))
{
    // Same O(1) exp. or statement
}.
```

$i$
$\downarrow$

$2$
$2^K$
$2^{K^2}$
$2^{K^3}$
$\vdots$
$2^{K^{\log_K(\log(n))}}$

where

$2^{K^{\log_K(\log n)}} \le n$ $\quad [m = \log_K \log_2 n]$

$2^{\log n} \le n$ $\quad$ No. of Iteration = $m$.

$n \le n$ $\quad$ Agree

So, there are in total $\log_K(\log(n))$ many iterations & each iteration take constant amt. of time.

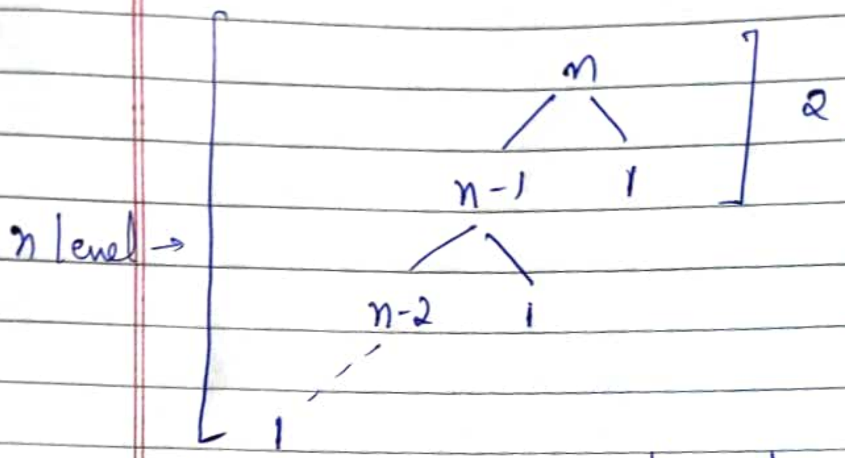
$\therefore T(n) = 1 + 1 + + \cdots \log_K \log n$ times.

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

Q7

Sol Given algo divides array in 99% & 1% part (Sorting Algo).

$\therefore T(n) = T(n-1) + O(1)$




n work is done at each level for merging.

$T(n) = (T(n-1) + T(n-2) + \cdots T(1) + O(1)) \times n$

$= n \times n$

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

Lowest Height $= 2$
Highest " $= n$

$\therefore$ Diff $= n - 2$  $n > 1$

- The given Algo provides linear result.

**Q** Arrange following in increasing order of rate of growth.

**Sol** For large values of $n$.

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

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

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