



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

$$\left. \begin{array}{l} j=1 \\ j=2 \\ j=3 \end{array} \right\} m \text{ levels}$$

$$\left. \begin{array}{l} i=1; \\ i=1+2; \\ i=1+2+3; \end{array} \right\}$$

$$\therefore 1 + 2 + 3 + \dots < n$$

$$1 + 2 + 3 + \dots + m < n$$

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

$$m \approx \sqrt{n}$$

$m \approx \sqrt{n}$
 by summation method
 $\Rightarrow \sum_{i=1}^m 1 \Rightarrow 1 + 1 + \dots + \sqrt{n}$ times
 $\therefore T(n) = \sqrt{n}$

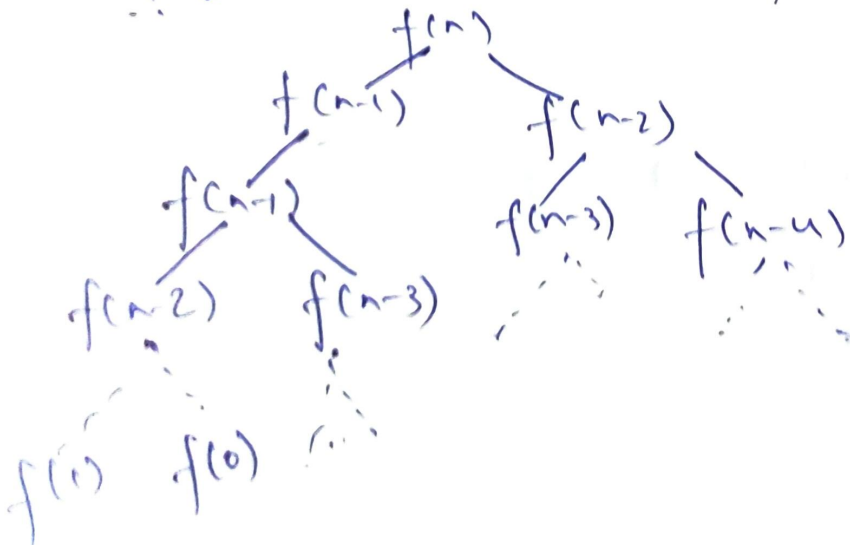


for fibonacci series:-

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

$$f(0) = 0$$

$$f(1) = 1$$



∴ at every function call we get two function calls for n levels:-

we have $\Rightarrow 2 \times 2 \dots n$ times

$$\therefore T(n) = 2^n$$

Maximum Space \div considering recursive stack:-

no of calls max. = n

For each call we have space complexity $O(1)$

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

3

a) $n \log n$:-

Quick sort

```
void func ( int arr[], int l, int h)
```

```
{ if ( l < h)
```

```
{ int pi = partition(arr, l, h);
```

```
  func ( arr, l, pi-1);
```

```
  func ( arr, pi+1, l h);
```

```
} }
```

```
int partition ( int arr[], int l, int h)
```

```
{ int pi = arr[h];
```

```
  int i = ( l-1);
```

```

for (int j = l; j <= h; j++)
{
    if (arr[i] < arr[j])
    {
        i++;
        swap(arr[i], arr[j]);
    }
}
swap(arr[i+1], arr[h]);
return (i+1);
}

```

⑥ n^3 :
Multiplication of two square matrix

```

for (i=0; i < n; i++)
{
    for (j=0; j < n; j++)
    {
        for (k=0; k < n; k++)
        {
            res[i][j] += arr[i][k] * arr[k][j];
        }
    }
}

```

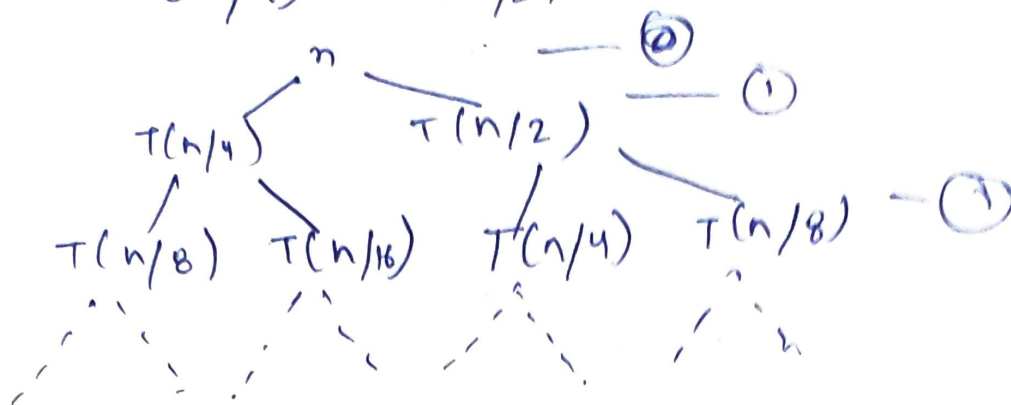
⑦ $\log(\log n)$

```

for (c=2; c < n; c = c * c)
{
    c++;
}

```

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



At level $\rightarrow 0 \rightarrow Cn^2$

$$1 \rightarrow \frac{n^2}{4^2} + \frac{n^2}{2^2} = \frac{C5n^2}{16}$$

$$2 \rightarrow \frac{n^2}{8^2} + \frac{n^2}{16^2} + \frac{n^2}{4^2} + \frac{n^2}{8^2} = \left(\frac{5}{16}\right)^2 n^2 C$$

$$\text{max level} = \frac{n}{2k} = 1$$

$$\Rightarrow k = \log_2 n$$

$$\therefore T(n) = \left(Cn^2 + \left(\frac{5}{16}\right)n^2 + \left(\frac{5}{16}\right)^2 \cdot \dots + \left(\frac{5}{16}\right)^{\log_2 n} n^2 \right)$$

$$T(n) = Cn^2 \cdot \left[1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \dots + \left(\frac{5}{16}\right)^{\log_2 n} \right]$$

$$T(n) = Cn^2 \times \left(\frac{1 - \left(\frac{5}{16}\right)^{\log_2 n}}{1 - \frac{5}{16}} \right)$$

$$= Cn^2 \times \frac{11}{5} \left(1 - \left(\frac{5}{16}\right)^{\log_2 n} \right)$$

$$\therefore T(n) = O(n^2 C) \Rightarrow O(Cn^2)$$

⑤

```

int fun (int n)
{
    for (i=1; i<=n; i++)
    {
        fun (j=1; j<=n; j+=i)
        {
        }
    }
}

```

for \rightarrow

i	j
1	1
2	1 + 3 + 5
3	1 + 4 + 7
4	1 + 5 + 9
\vdots	\vdots
n	

$$\sum_{i=1}^n \frac{(n-1)}{i}$$

$$\therefore T(n) = \frac{(n-1)}{2} + \frac{(n-1)}{2} + \frac{(n-1)}{3} + \dots + \frac{(n-1)}{n}$$

$$T(n) = n \left[1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \right] = 1 \times n \left[1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \right]$$

$$= n \log n = \log n$$

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

⑥

```

for (i=2; i<=n; i = pow(i, k))
{
    O(1)
}

```


$$\text{for } \rightarrow \begin{matrix} i \\ 2^1 \\ 2^k \\ 2^{k^2} \\ 2^{k^3} \\ \vdots \\ 2^{k^m} \end{matrix}$$

$$\text{when } 2^{k^m} \leq n$$

$$k^m = \log_2 n$$

$$m = \log_k \log_2 n$$

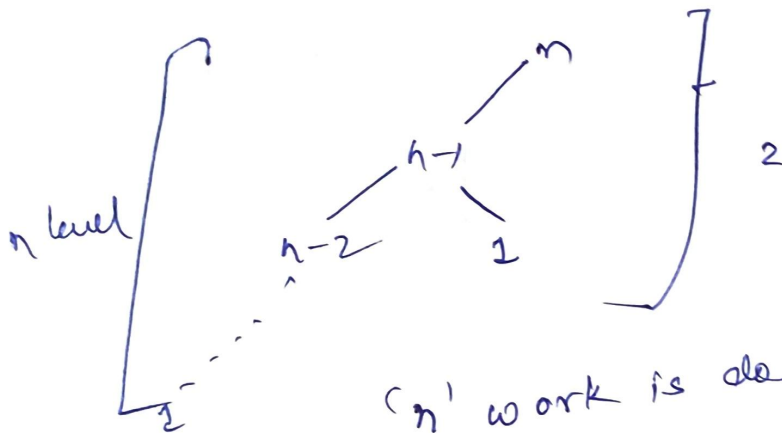
$$\therefore \sum_{i=1}^m 1$$

$$\Rightarrow 1 + 1 + \dots + 1 \text{ } m \text{ times}$$

$$\Rightarrow T(n) = O(\log_k \log_2 n)$$

⑦ Given algo divides array in 99% & 1% part

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



(n) work is done at each level for merging

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

$$= n \times n$$

$$T(n) = O(n^2)$$

$$\begin{matrix} \text{lowest} & \text{higher} = 2 \\ \text{height} & \text{higher} = n \end{matrix}$$

$$\therefore \text{diff} = n-2 \quad \therefore (n > 1)$$

⑧ considering for large values of 'n'.

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

$$b) 1 < \log \log n < \sqrt{\log n} < \log n < \log 2n < 2 \log n \\ < n < n \log n < 2n < 4n < \log(n!) < n^2 < n < 2 \\ \log 2n < 5n$$

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