Tutorial- 2

Name → Janvi Sah
Section → G
Roll No. → 62
University Roll No. → 2016794

Ans 1.

$$\left.\begin{array}{ll} j=1 & i=1 \\ j=2 & i=1+2 \\ j=3 & i=1+2+3 \end{array}\right] m\text{ - level}$$

for(i)

$\because$ $1+2+3+ \ldots + < n$

$1+2+3+m < n$

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

$m \approx \sqrt{n}$

By summation method

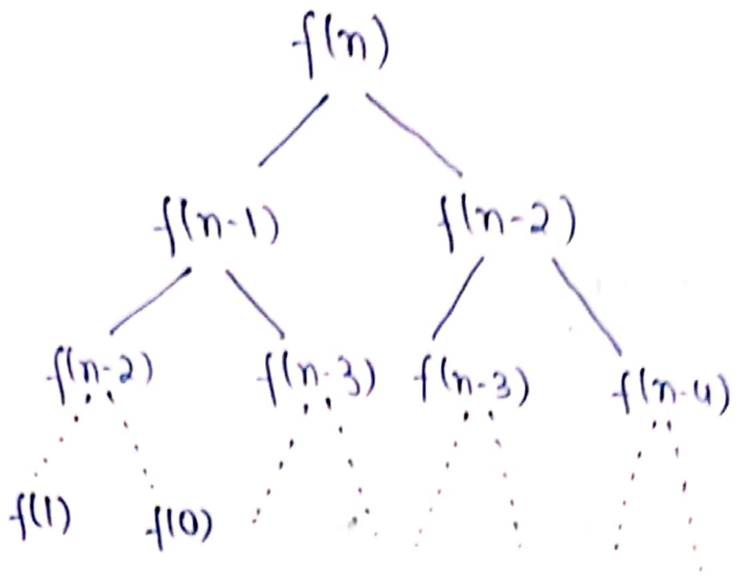
$\Rightarrow \sum_{i=1}^{m} 1 \Rightarrow 1+1+1 \ldots + \sqrt{n}$ times.

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

Ans 2. For fibonacci series

$f(n) = f(n-1) + f(n-2)$ $\qquad f(0) = 0$

By forming a tree $\qquad f(1) = 1$



At every func. call we get 2 function calls

∴ for n levels

We have → 2 × 2 . . . n times

∴ $\boxed{T(n) = 2^n}$

Maximum Space Considering Recursion Stack:

No. of calls maximum = n

For each call we have space complexity O(1)

∴ $T(n) = O(n)$

Without considering Recursion stack:

each call we have time complexity O(1)

∴ $\boxed{T(n) = O(1)}$

Ans 3.

(i) n logn → Quick Sort

```
void quicksort(int arr[], int low , int high)
{
  if (low < high)
  {
    int pi = partition(arr, low ,high);
    quicksort(arr, low, pi - 1);
    quicksort (arr, pi+1, high);
  }
}

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

  swap (& arr[i+1], & arr [high]);
  return (i+1);
}
```

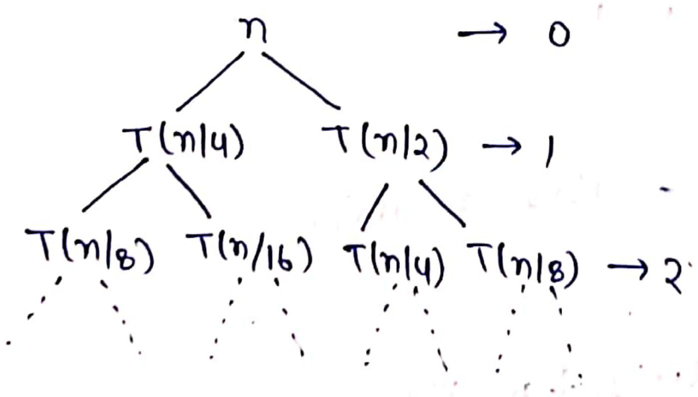
(ii) $n^3 \rightarrow$ Multiplication of 2 square matrix

```
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];
      }
```

(iii) $\log(\log n)$

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

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



At level

$0 \rightarrow Cn^2$

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

Scanned with CamScanner

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

$\vdots$

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

$\Rightarrow k = \log_2 n$

$$T(n) = C(n^2 + (5/16)n^2 + (5/16)^2 n^2 + \dots + (5/16)^{\log n} n^2)$$

$$T(n) = Cn^2 [1 + (5/16) + (5/16)^2 + \dots + (5/16)^{\log n}]$$

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

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

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

$$\boxed{O(Cn^2)}$$

Ans 5.

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

$j = (n-1)/i$ times

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

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

$$T(n) = n[1 + 1/2 + 1/3 + \dots + 1/n] - 1 \times [1 + 1/2 + 1/3 + \dots + 1/n]$$

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

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

Scanned with CamScanner

Ans 6.

$\rightarrow$ for $\quad i \qquad$ where

2^1

$2^k \qquad\qquad 2^{k^m} <= n$

$2^{k^2} \qquad\qquad k^m \leq \log_2 n$

$2^{k^3} \qquad\qquad m = \log k \log_2 n$

$\vdots$

2^{k^m}

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

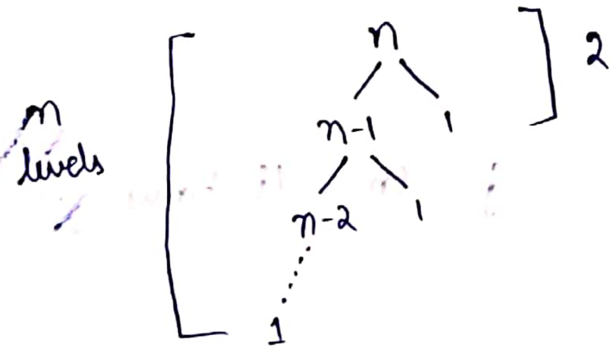
$1 + 1 + 1 + \dots$ m times

$$\boxed{T(n) = O(\log_k \log n)}$$

Ans 7.

Given algorithm divides array in 99% and 1% part

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



'n' work is done at each level

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

$T(n) = n \times n$

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

lowest height = 2

highest height = n $\qquad$ linear result

$\therefore$ difference = n - 2 $\quad$; n > 1

Ans 8

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

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