Name:- Ishu Verma
Section:- CST SPL 2
Roll no:- 04

# Tutorial 2

Q1.)
```
void fun (int n)
{   int j=1, i=0;
  while (i<n)
  {   i+=j;
      j++;
  }
}
```

$j=1 \qquad i=1$

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

$j=3 \qquad i=1+2+3$

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

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

$\frac{m^2+m}{2}<n \Rightarrow m^2<\sqrt{n}$

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

By Summation method.

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

$= \sqrt{n}$

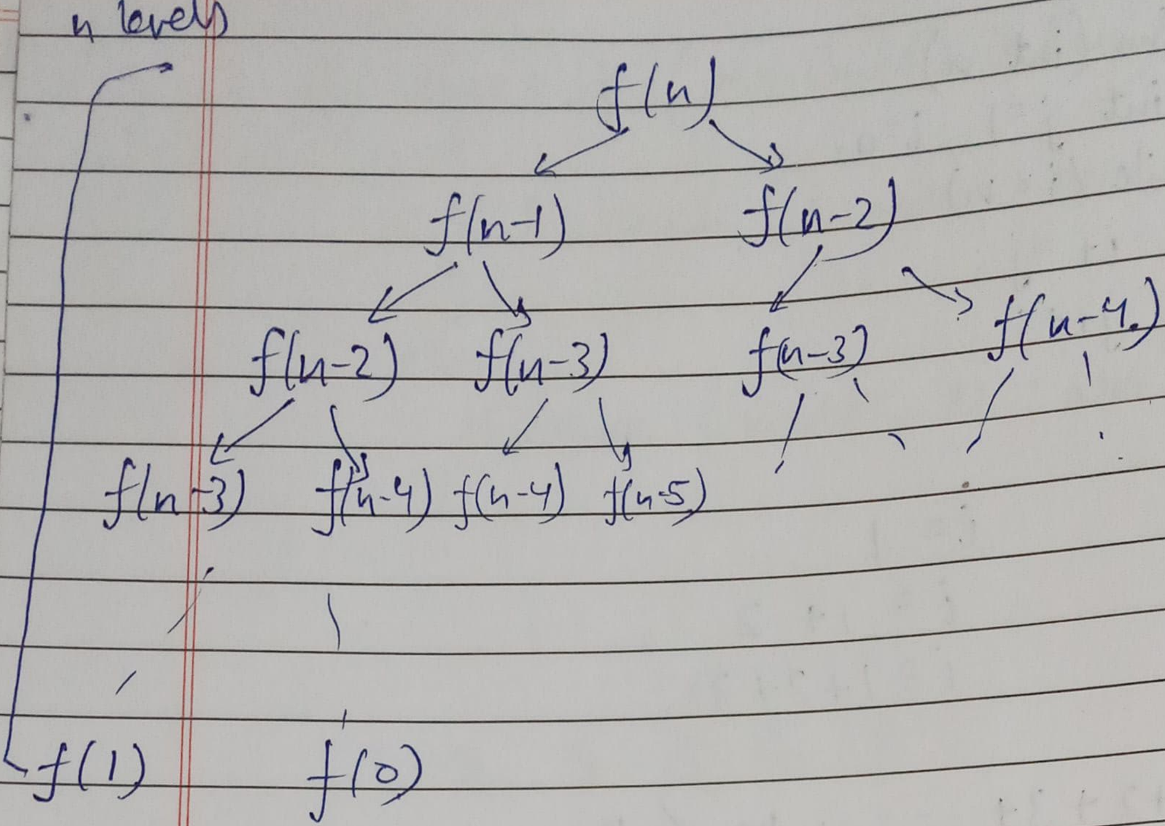
$T(n)=\sqrt{n}$ Ans

Q2.) For fibonacci series

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

$f(0)=0 \quad f(1)=1$

Forming a tree




At every function call, we get 2 function calls
For n levels, $2 \times 2 \cdots n$ times.
$= 2^n$

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

Maximum space :- Space complexity depends on the maximum depth of the tree so
Space complexity $= O(n)$.

Q3.) $T(n) = O(n^3)$

int ~~t = 8~~
~~vector <int, int>~~

Multiplication of two square matrix.

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

nlogn

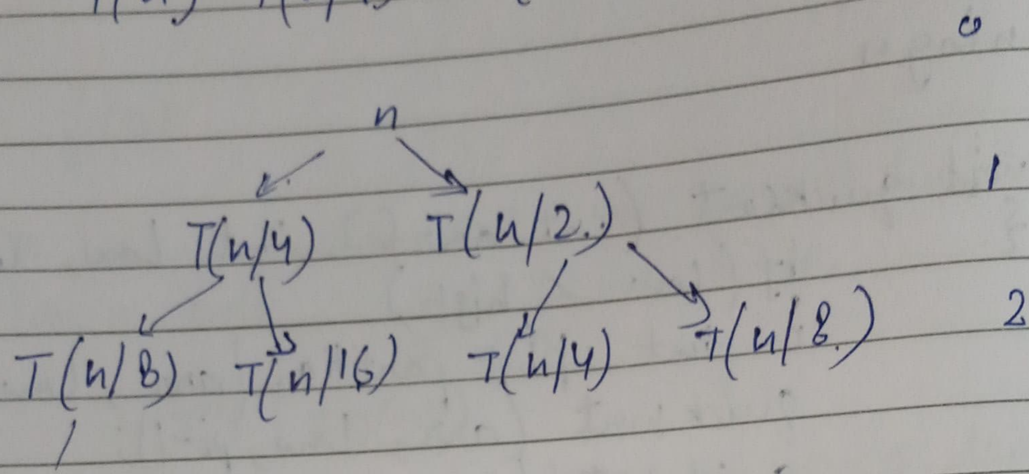
```
void quicksort (int arr[], int low, int heigh)
{
    if(low < high)
    {   int pi = partition (arr, low, high);
        quicksort (arr, low, pi-1);
        quicksort (arr, pi+1, high);
    }
}
int partition (int arr[], int low, int high)
{
    int pivot = arr[high];
    int i = (low-1);

    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);
}
```

log(log n.)

```
for(i=2; i<n; i=i*i)
{
    ~~for (j=0; j<c2; j++)~~
    count ++;
}
```

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




At level 0 $\longrightarrow$ $cn^2$

1 $\longrightarrow$ $\frac{n^2}{4^2} + \frac{n^2}{2^2} = \frac{c5n^2}{16}$

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

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

$\Rightarrow k = \log n$

$T(n) = c\left(n^2 + \frac{5}{16}n^2 + \left(\frac{5}{16}\right)^2 n^2 ----\right)$

$= \cancel{\left(\left(\frac{5}{16}\right)^{\log n} n^2\right)}$

$$= cn^2\left[1+\frac{5}{16}+\left(\frac{5}{16}\right)^2+\cdots\left(\frac{5}{16}\right)^{\log n}\right]$$

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

$$= cn^2 \frac{16}{\cancel{16}11}\left(1-\left(\frac{5}{16}\right)^{\log n}\right)$$

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

Q5.)

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

| i | j |
|---|---|
| 1 | 1 |
| 2 | 1+3+5 |
| 3 | 1+4+7 |
| ⋮ | 1+5+9 |
| | ⋮ |
| n | |

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

$$T(n) = \frac{n-1}{1}+\frac{n-1}{2}+\cdots+\frac{n-1}{n}$$

$$= n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots\frac{1}{n}\right)-1\left[1+\frac{1}{2}+\frac{1}{3}+\cdots\frac{1}{n}\right]$$

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

Q6.) for $(i=2; i<=n; i = pow(i,k))$
$\{ // O(1)$
$\}$

where $k$ is constant

$$T.C = 2, 2^k, 2^{k^2}, 2^{k^4} \cdots 2^{k\log_k(\log n)}$$

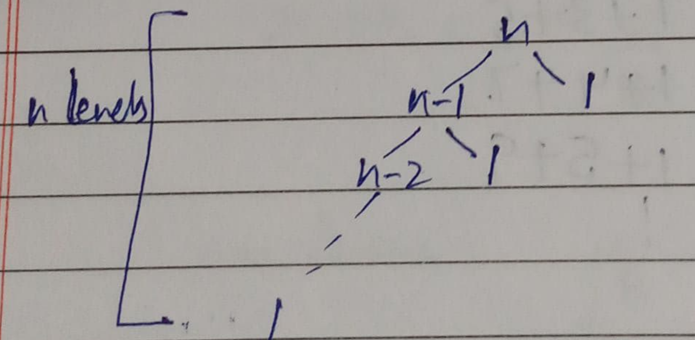
$$2^{k\log_k(\log n)} = 2^{\log n} = n$$

so there are total $\log_k(\log n)$ iterations.

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

Q7.) Given algo divides array in 99% and 1% part.

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




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

Lowest height = 2.
~~heighte~~ highest height = n.
diff = n-2 $\quad n>1$
The given algorithm provides linear result.

Q8.) Considering 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.) $96 < \log_8 n < \log 2n < 5n < n\log_6 n < n\log_2 n < \log(n!) < 8n^2 < 7n^3 < n! < 8^{2n}$