MUKUL Rawat
42 - CST

Page No. ____
Date

① (Tutorial-2)

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

for
$j=1$ $i=1;$
$j=2$ $i=1+2;$
$j=3$ $i=1+2+3;$ m levels
⋮

for (i)
$\because \; 1+2+3+\dots < n$

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

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

$m \approx \sqrt{n}$

$\because$ by summation method

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

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

Mukul Rawat

2

Page No.

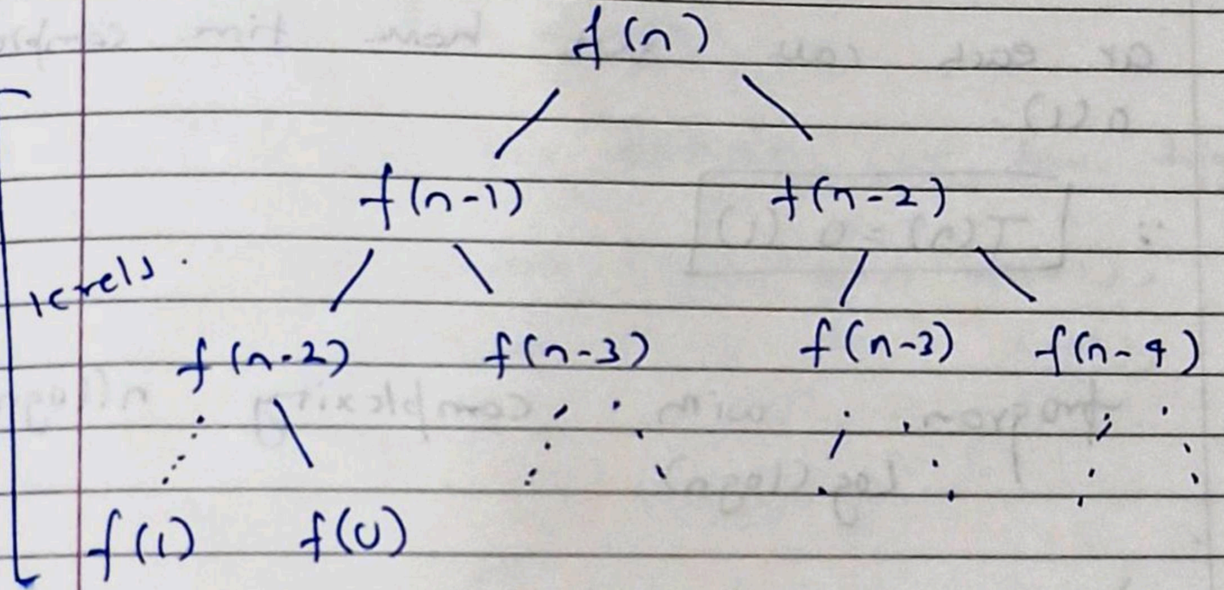
Date

(Q2.) For fibonacci series:-

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

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

by forming tree :-



∴ At every function call we get 2 function calls

∴ for n levels :-

we have = $2 \times 2 \cdots$ n times

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

Maximum space:-

considering recursive stack:-

no of calls maximum $= n$

③ Mukul Rawat

Page No.
Date

Q3) for each call we have space complexity O(1)

∴ $\boxed{T(n) = O(n)}$

Without considering recursive stack :-

at each call we have time complexity O(1)

∴ $\boxed{T(n) = O(1)}$

4

Mul Rawat

Q3)

① nlogn

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);
}
```

n^3,

5

Rawat

Page No.

Date

(2) n^3

Multiplication of two sq. 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=1;  i<n;  i=i*2)
   {
      for (j=i ; j<n  ; j=j*2)
        {
           count++;
        }
   }
```

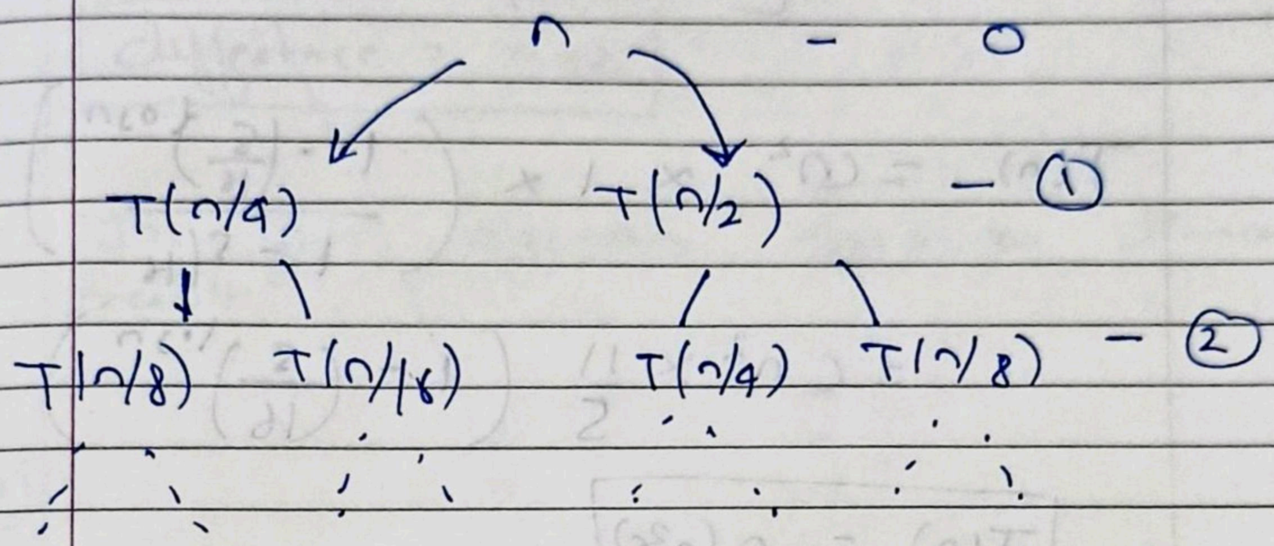
```
for  (i=2;  i<n;  i= i*i)
      {
        count ++;
      }
```

6

Mukul Rawat

Page No.
Date

Q4.) $T(n) = T(n/4) + T(n/2) + c \times n^2$



At level :-

$0 \rightarrow cn^2$

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

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

⋮

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

$\Rightarrow k = \log_2 n$

$\therefore T(n) = c\left(n^2 + (5/16)n^2 + \left(\frac{5}{16}\right)^2 n^2 \dots + (5/16)^{\log n} n^2\right)$

(7) [signature]

Page No.
Date

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

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

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

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

Q5.)

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

}
```

for

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

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

Mrunal Tiwari

Page No.
Date

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

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

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

$$= n\log n - \cancel{n\log n}\ \log n$$

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

Q6.)

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

for i

2^1

2^k

2^{k^2}

2^{k^3}

$\vdots$

2^{k^m}

where, $2^{k^m} <= n$

$k^m = \log_2 n$

$m = \log_k \log_2 n$

3

Mant Rawat

Page No.

Date

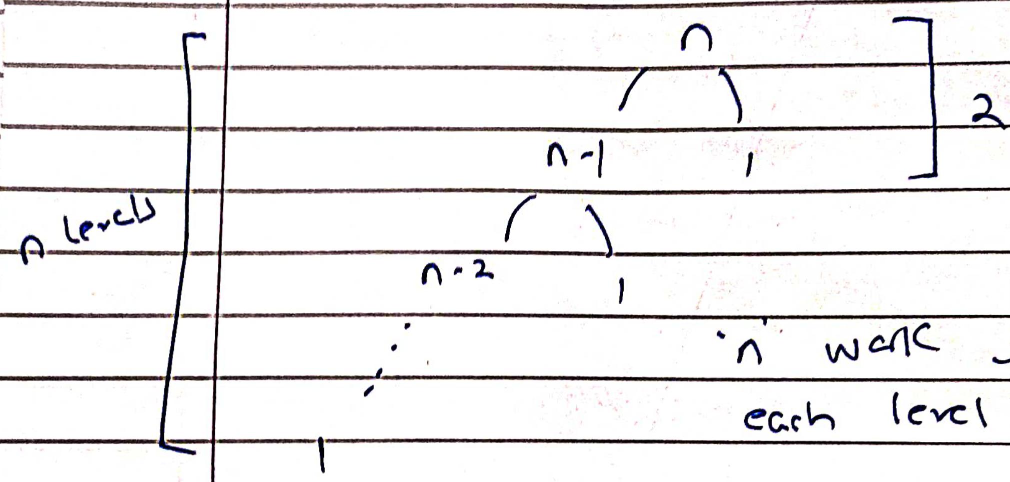
$\because \sum_{i=1}^{m} 1$

$\Rightarrow 1+1+1\ldots$ m times

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

Q7.) Given algo divides array in 99% & 1% part (sorting algo)

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



'n' work is done at each level for merging

$T(n) = \left(T(n-1) + T(n-2) + \ldots T(1) + O(1)\right) \times n$

$= n \times n$

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

lowest height = 2

highest " = n

$\boxed{\because \text{diff.} = n-2} \quad n>1$

Page No.
Date

The given algorithm produces linear result.

Q8.) Considering for large value of 'n'.

<100

a.) $< \log\log n < \log n < (\log n)^2 <$

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

$2^n < 4^n < 2^{2^n}$

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.) $\log 2n < 5n$

$96 < \log_8 n < \cancel{\log_8 n} < n\log_6 n < n\log_2 n$

$\log(n!) < 8n^2 < 7n^3 < n! < 8^{2n}$