Assignment → 2

Name → Yashraj Daiyan
Rollno. → 39 / CST

Tutorial - 2

Ans-1

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

Time Complexity → $O(\sqrt{n})$

1st time $i=1$
2nd time $i=3$ $(i=1+2)$
3rd time $i=6$ $(i=1+2+3)$
⋮
k^{th} time $i = \frac{k(k+1)}{2} = x^2 < n$

$x = \sqrt{n}$

Ans-2 * $fib(n) = fib(n-1) + fib(n-2)$

```
fib(n): if n<=1
            return 1
        return fib(n-1) + fib(n-2)
```

Time Complexity

$T(n) = T(n-1) + T(n-2) + c$
$= 2T(n-2) + c$

{ let $(T(n-1) \simeq T(n-2)$ }

putting $n = n-2$

$T(n-2) = 2^*(2T(n-4)+c)+c$
$= 4T(n-4) + 3c$ ——— (1)

putting $n = n-4$

$T(n-4) = 2^*(4T(n-4)+3c)+c$
$= 8T(n-4) + 7c$
$= 2^3 \times T(n-2k) + (2^k - 1)c$

Yashraj

$n-2k=0 \Rightarrow n=k$

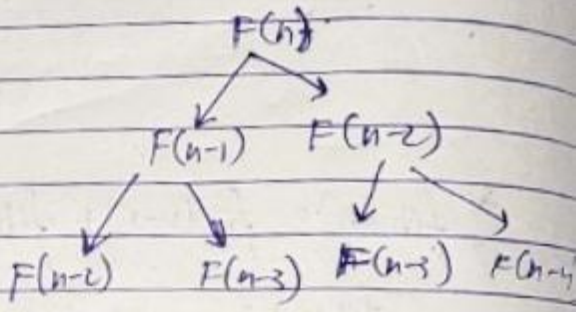
$T(n) = 2^n * T(0) + (2^n-1)C$

$2^n * 1 + 2^n C - C$

$2^n(1+C) - C \simeq 2^n$ // Constant

$O(2^n)$

Space Complexity :- $O(N)$



Ans-3 Merge Sort → nlogn

⇒ for time complexity → n^3 ; we can use three nested loops → O(n

```
for (int i=0; i<n; i++)
  { for (int j=0; j<n; j++)
    { for (int k=0; k<n; k++)
        { some O(1) expression }
    } }
```

⇒ for time complexity → log(logn)
we can use the following function

```
for (int i=2; i<n; i= pow(i,c))
    { Some O(1) expression }
```

where k is Constant.

⇒ for time complexity $n \log n$
we can use the following function

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

Ans-4 $T(n) = T(n/4) + T(n/2) + cn^2$ $\{T(\frac{n}{2}) \geq T(n/4)\}$

using masters method

$$T(n) = aT(n/b) + f(n)$$

$a \geq 1, b > 1, c = \log_b a$ Comparing n^c & $f(n)$

We get, $c = \log_2 2 = 1$

$f(n) > n^c$

$T(n) = \Theta(f(n))$

$\Rightarrow \Theta(n^2)$

Ans-5 for $i=1 \longrightarrow j = 1, 2, 3, 4, \ldots n$ (Sum for n times)

for $i=2 \longrightarrow j = 1, 3, 5, \ldots n/2$ (Sum for $n/2$ times)

for $i=3 \longrightarrow j = 1, 4, 7, \ldots n/3$ (Sum for $n/3$ times)

$T(n) = n + n/2 + n/3 + n/4 + \ldots$

$= n(1 + 1/2 + 1/3 + 1/4 + \ldots)$

$= n \int_1^n 1/x \Rightarrow n \int_1^n \frac{dx}{x} \Rightarrow [\log x]_1^n$

$n \log n$

Time Complexity of following function is $n \log n$

Yashvi

Ans 1 For first iteration $i = 2$

2^{nd} " $i = 2^k$

3^{rd} " $i = (2^k)^k = 2^{k^2}$

⋮

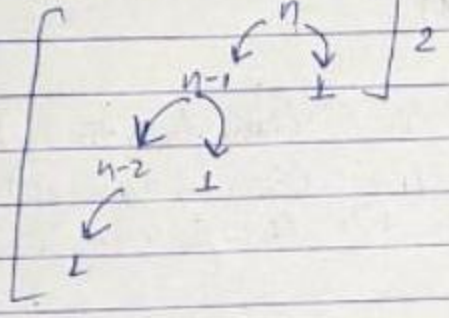
n^{th} " $i = 2^{k^i}$; loop ends at $2^{k^i} = n$

apply log $\log n = \log 2^{k^i} \Rightarrow k^i = \log n$

apply log again $\log(k^i) = \log\log n \rightarrow i = \log_k(\log n)$

Ans 2 The given algorithm divides array in 99% & 1% part

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



'n' work done at each level for merging

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

$= n \times n$

$= O(n^2)$

Lowest height = 2

Highest height = n

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

given algorithm produces linear result.

Yes

Ans-8 Considering for large value of 'n':

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

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

Yaman