VANSH KALRA
CST
2S
DAA 2017576

DATE
PAGE

Tutorial-2

Q1. What is the time complexity of below code and how?

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

A1. Values after execution of while loop

Ist time	$i = 1$
2nd time	$i = 3 = 1+2$
3rd time	$i = 6 = 1+2+3$
4th time	$i = 10 = 1+2+3+4$

let for ith time $i = (1+2+3+\dots i) < n$

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

$$= i^2 < n$$

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

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

Q2. Write recurrence relation for the recursive function that prints fibonacci series. Solve the recurrence relation to get complexity of the program. What will be the space complexity of this program & why.

A2. $fib(n) = fib(n-1) + fib(n-2)$

```
int fib (int n)
{
    if (n<=1)        — O(1)
```

DATE

PAGE

return n;

return fib(n-1) + fib(n-2); — $T(n-1) + T(n-2)$

}

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

fib(n)

fib(n-1) fib(n-2)

$\cancel{F(n) = 2F(n-2)+1}$ $\cancel{[\text{let } T(n-1) \approx T(n-2)]}$

$\cancel{T(n-2) = 2*(2F(n-2-2)+)+1}$

$\cancel{= 2*(2F(n}$

fib(n) — 1

fib(n-1) fib(n-2) — 2

fib(n-2) fib(n-3) fib(n-3) fib(n-4) — 4

$1+2+4+8+\ldots$

$a=1 \quad r=2$

$$\Rightarrow \frac{a(r^{\text{terms}}-1)}{r-1}$$

$$= 2^{\text{terms}} - 1$$

$$= 2^{n+1} - 1$$

$$\Rightarrow 2^{n+1}$$

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

There is one entry in stack at every function call and it remains inside stack till it returns the value. Maximum entry at any instance $<= n$.

$\boxed{\therefore \text{Space Complexity} = O(n)}$

Q3. Write programs which have complexity - $n \log n$, n^3, $\log(\log n)$.

- n logn

```
for (i=1; i<=n; i*=2)
{
    for (j=1; j<=n; j++)
        sum = sum+i;
}
```

i	j
1	n
2	n
4	n
:	:
logn	n

$T = O(n \log n)$

DATE

PAGE

- $\underline{n^3}$

```
for (i=0; i<n; i++)
{
    for (j=0; j<n; j++)
    {
        for(k=0; k<n; k++)
            sum+=k;
    }
}
```

- $\underline{\log \log n}$

```
for (j=1; j<n; j*=2)
{
    for (k=j; k>=1; k/=2)
        sum+=j;
}
```

Q4. Solve the following recurrence relation $T(n) = T(n/4) + T(n/2) + cn^2$

A4. $T(n) = 2T(n/2) + cn^2$ $\qquad T(n/2) \geq T(n/4)$

Using masters method

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

$a \geq 1 \quad b \geq 1 \quad c = \log_b a$

Comparing n^c & $f(n)$ we get

$$c = \log_2 2 = 1$$

$$f(n) > n^c$$

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

DATE
PAGE

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

Q5 What is the time complexity of following function fun()?

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

As.

for	i=1	j=1,2,3,4,.... n	(run for n times)
for	i=2	j=1,3,5....	(run for n/2 times)
for	i=3	j=1,4,7....	(run for n/3 times)

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

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

$= n \int_1^n \frac{1}{x}$

$= n \log x \,/_1^n$

$= n \log n$

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

Q6 What should be the time complexity of

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

where k is a constant.

DATE

PAGE

A6. I^{st} iteration $i = 2$

II^{nd} iteration $i = 2^k$

III^{rd} iteration $i = 2^{k^2}$

⋮

n^{th} iteration $i = 2^{k^i}$

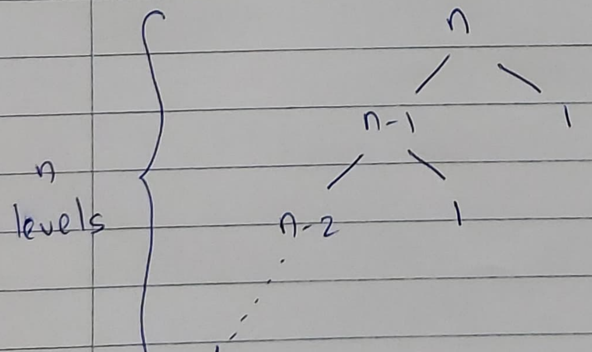
$$n = 2^{k^i}$$
$$\log n = \log 2^{k^i}$$
$$\log n = k^i \log 2$$
$$\log \log n = i \log k$$
$$i = \log_k \log n$$

$$\boxed{T = \log_k \log n}$$

Q7. Write a recurrence relation when quick sort repeatedly divides the array into two parts of 99% & 1%. Derive the time complexity in this case. Show the recursion tree while deriving time complexity & find the difference in heights of both the extreme parts. What do you understand by this analysis?

A7.

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



$T(n) = [T(n-1) + T(n-2) + \dots + T(1) + O(1)] \times \underset{\substack{\uparrow \\ \text{for merging}}}{n}$

$T(n) = n \times n$

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

Lowest height = 2
Highest height = n

Difference b/w highest & lowest heights = $n-2$ $\quad n > 1$

Analysis - The given algorithm provides linear result in the form of sorted array.

DATE

PAGE

Q8. Arrange the following in increasing order of rate of growth.

a) n, n!, logn, loglogn, root(n), log(n!), nlogn, log^2(n), 2^n, 2^(2^n), 4^n, n^2, 100

b) 2(2^n), 4n, 2n, 1, logn, log(log(n)), $\sqrt{\log n}$, log2n, 2logn, n, log(n!), n!, n2, n logn.

c) 8^(2n), $\log_2 n$, $n\log_6 n$, $n\log_2 n$, log(n!), n!, $\log_8(n)$, 96, 8n2, 7n3, 5n.

A8. a) $100 < \log(\log n) < \log n < \log^2 n < root(n) < n < n\log n < n^2 < 2^n < 4^n < 2^{2^n} < \log(n!) < n!$

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

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