Saathi

Date __/__/__

Tutorial - 2

Ques:1 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++;  }
}
```

Sol^n

$$i = \overbrace{0, 1, 3, 6, 10, 15, 21, \ldots n}^{k\text{ - terms}}$$

let the Sum of above k terms is S_K

$$S_K = 1+3+6+10+15+21, \ldots + T_K \quad \text{— (1)}$$

$$S_{K-1} = 1+3+6+10+15+21, \ldots + T_{K-1} \quad \text{— (2)}$$

Subtracting (2) from (1)

$$T_K = S_K - S_{K-1} = 1+2+3+4+5+6+ \ldots + K$$

we have $T_K = n$

$\therefore$ $1+2+3+4+5+ \ldots + K = n$

$$\frac{K(K+1)}{2} = n \Rightarrow K^2 + K - 2n = 0$$

$$\Rightarrow K = \frac{-1 \pm \sqrt{8n+1}}{2}$$

taking only positive value we get total no. of times the loop runs for $i = K+1 = \frac{\sqrt{8n+1}}{2}$

Page No.

40

Time Complexity, $T(n) = O\left(\frac{\sqrt{8n+1}}{2}\right)$

$= O(\sqrt{n})$

Ques 2 Write Recurrance Relation for the recursive function that prints fibonacci series. Solve the recurrance relation to get time complexity of the program. what will be space complexity of this program and why?

Soln:

Recursive function:

```
int fib(int n)
{ if (n<=1)              -> O(1) = C
     return n;
  return fib(n-1) + f(n-2)  -> T(n-1)+T(n-2)
}
```

Recurrance Relation, $T(n) = T(n-1) + T(n-2) + C$

$T(n-1) \simeq T(n-2)$

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

$T(n-2) = 2*(2T(n-2-2)+C) + C$

$= 4T(n-2) + 3C$

$T(n-4) = 2*(4T(n-2)+3C) + C$

$= 8T(n-3) + 7C$

Page No.

Generalising

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

put $n - K = 0$

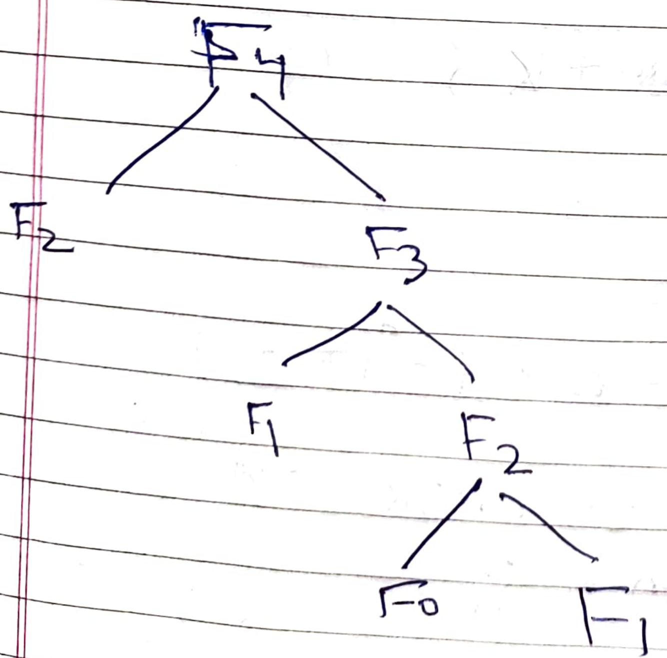
$n = K$

put $n = K$

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

Time Complexity = $O(2^n)$

Space Complexity : Space is proportional to the maximum depth of the recursion tree.



Hence Space Complexity of fibbonacci recursion is $O(N)$

Date __/__/__

Ques 3 Write programs which have Complexity

Soln

1. $n(\log n)$

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

2. n^3

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

3. $\log n(\log n)$

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

Date

Ques 4 Solve the Recurrence Relation

$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + cn^2$$

Soln

$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + cn^2$$

$\therefore$

$$T\left(\frac{n}{4}\right) \simeq T\left(n/2\right)$$

$$\Rightarrow T(n) = 2T\left(\frac{n}{2}\right) + cn^2$$

As $a \geq 1$ and $b > 1$

$\therefore$ using master's Method,

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

$$c = \log_b a$$

$$c = \log_2 2 = 1$$

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

$\therefore$

$$T(n) = O(f(n))$$

$$= O(n^2)$$

Ques5 What is the time complexity of the following function.

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

Sol^n

for $i=1$, j is $1, 2, 3, 4 \ldots$ run n-time

for $i=2$, j is $1, 3, 5 \ldots$ upto $n/2$ times

for $i=3$, j is $1, 4, 7, \ldots$ run for $n/3$ times

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

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

$$= n\int_1^n dx/x$$

$$= [\log x]_1^n$$

$\Rightarrow$ Time complexity $= n\log n$

Date __/__/__

Ques 6: What should be the time complexity of

```
for (int i = 2; i <= n; i = pow(i, k))
{
    some O(1) expression or statements
}
```

where k is a constant.

Sol^n for first iteration $i = 2$

Second iteration $i = 2^k$

third iteration $i = (2^k)^k = 2^{k^2}$

$\vdots$

nth iteration, $i = 2^{k^i}$ loop ends at $2^{k^i} = n$

apply $\log n = \log_2 k^i$

$k^i = \log n$

$i = \log_k (\log n)$

Ques 7 Write a recurrence relation when quick sort repeatedly divides the array in to two parts of 99% and 1%. Derive the time complexity and find the difference in heights of both the extreme parts. What do you understand by this analysis.

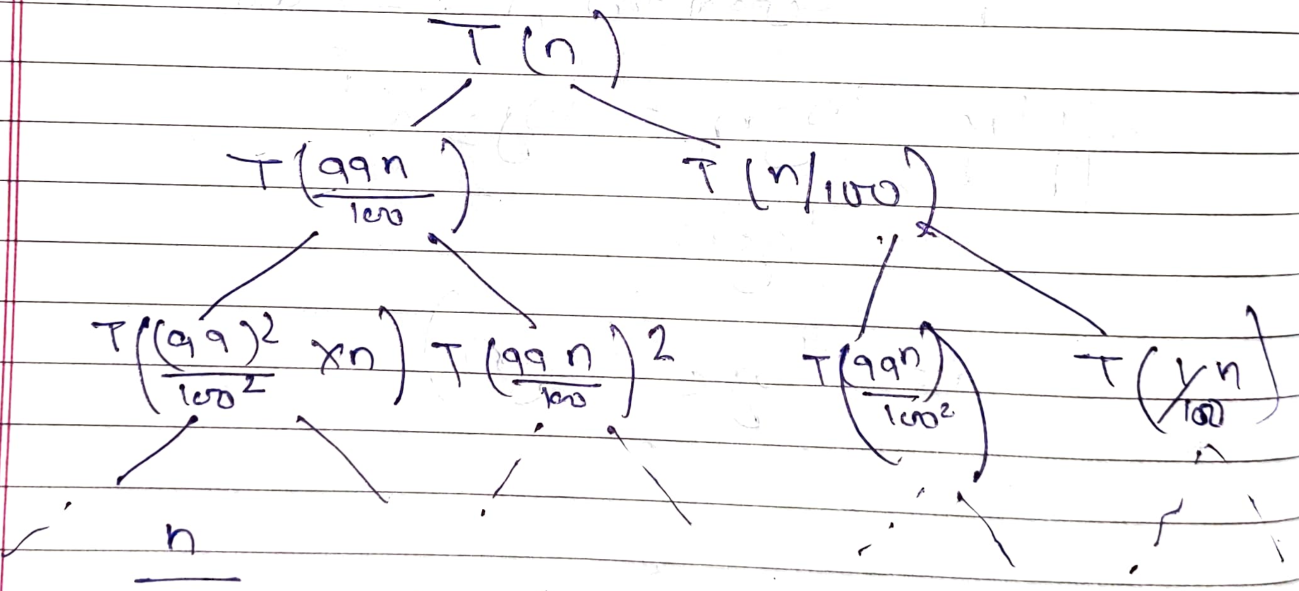
Soln

99 to 1 in quick sort

When pivot is where from front or end always

So, $T(n) = T(99n/100) + T(n/100) + O(n)$

$$T(n) = T(99n/100) + T(n/100) + O(n)$$



$$n = (99/100)^K$$

$$\log n = K \log 99/100$$

$$K = \log_{\frac{100}{99}} n$$

$\therefore$

$$T.C = n * \log_{\frac{100}{99}}(n)$$

Date ___/___/___

Ques 8 Arrange the following in increasing order of rate of growth.

Sol^n →

a) $100 < \log\log(n) < \log^2 n < \log n < \log n! < n < n\log n < n^2 < 2^n < 4^n < 2^\wedge(2^\wedge n) < n!$

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