Gaurav Sengar
Roll No-03

Tutorial-2

Date.....

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++; }
}
```

Soln:

$i = \underbrace{0, 1, 3, 6, 10, 15, 21, \ldots}_{K\text{-terms}} \ldots n$

let the sum of above K terms is S_K

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

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

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 $i = K+1 = \frac{\sqrt{8n+1}}{2}$

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

Teacher's Sign.....

Date.....

Qu.2 Write Reurrance Relation for the recursive function that prints fibonacci series. Solve the reurrance relation to get time complexity of the program. What will be time space complexity of this program and why?

Sol^n:

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)
}
```

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

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

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

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

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

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

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

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

Put $n-K = 0$

$n = K$

Put $n = K$

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

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

$= 2^n (1+C) - C$

$= 2^n$

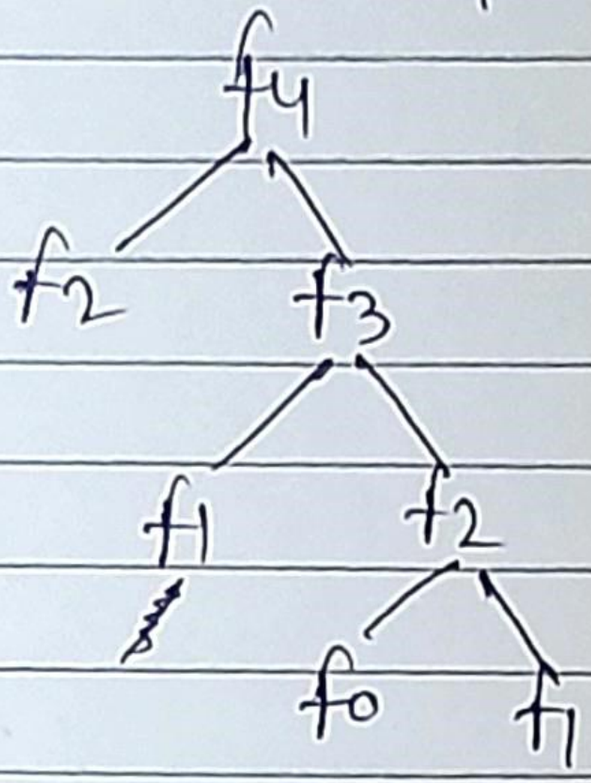
Time Complexity $= O(2^n)$

Spiral

Teacher's Sign.....

Date

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



Hence space complexity of Fibonacci 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.....

Qu. 4: Solve the Recurrance Relation $T(n) = T(n/4) + T(n/2) + cn^2$

Sol^n

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

$$\therefore T\left(\frac{n}{4}\right) \leq T\left(\frac{n}{2}\right)$$

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

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

$\therefore$ Using master's Method.

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

$$c = \log_b a$$

$$c = \log_2 2 = 1$$

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

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

$$= O(n^2)$$

Qus 5: 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
    }}
```

Teacher's Sign.....

Date.....

Sol^n)

for $i=1$, j is 1,2,3,4 ------ run for n-times

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

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

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

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

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

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

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

Qu-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 $i = (2^k)^k = 2^{k^2}$

!

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

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

$$k^i = \log n$$

$$i = \log_k (\log n)$$

Date

Que-7 Write a recurrence relation when quick sort repeatedly divides the array into two parts of 99% and 1%. Derive the time complexity in this case. Show the recursion tree while deriving time complexity and find the difference in heights of both the extreme leafs. What do you understand by this analysis.

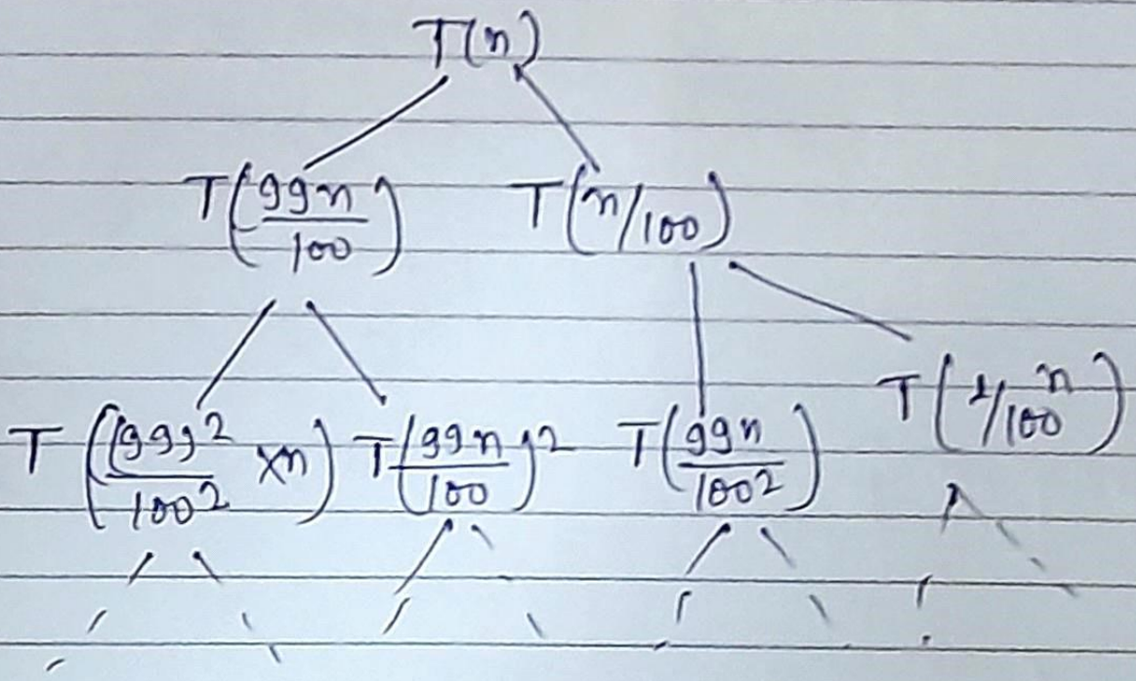
Sol^n

99 to 1 in quick sort
when pivot is where from front or end always.

So,

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

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



$\frac{n}{}$

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

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

$$k = \log n \frac{100}{99}$$

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

Spiral

Teacher's Sign

Date.....

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

Soln.

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

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