ASSIGNMENT- (TIME & SPACE COMPLEXITY)

Q1. Analyze the time complexity of the following Java code and suggest a way to improve it:

```
int sum = 0;
for(int i = 1; i <= n; i++) {
  for(int j = 1; j <= i; j++) {
  sum++;
}
}
```

Ans- Time Complexity for the following code will be- $O(n^2)$

Q2. Find the value of T(2) for the recurrence relation T(n) = 3T(n-1) + 12n, given that T(0) = 5.

Ans- Given, T(0)=5

Now,

T(1)= 3T(0)+12*1

T(1)=3*5+12 [T(0)=5]

T(1)=27

Therefore,

T(2)=3T(2-1)+12*2

T(2)=3T(1)+24

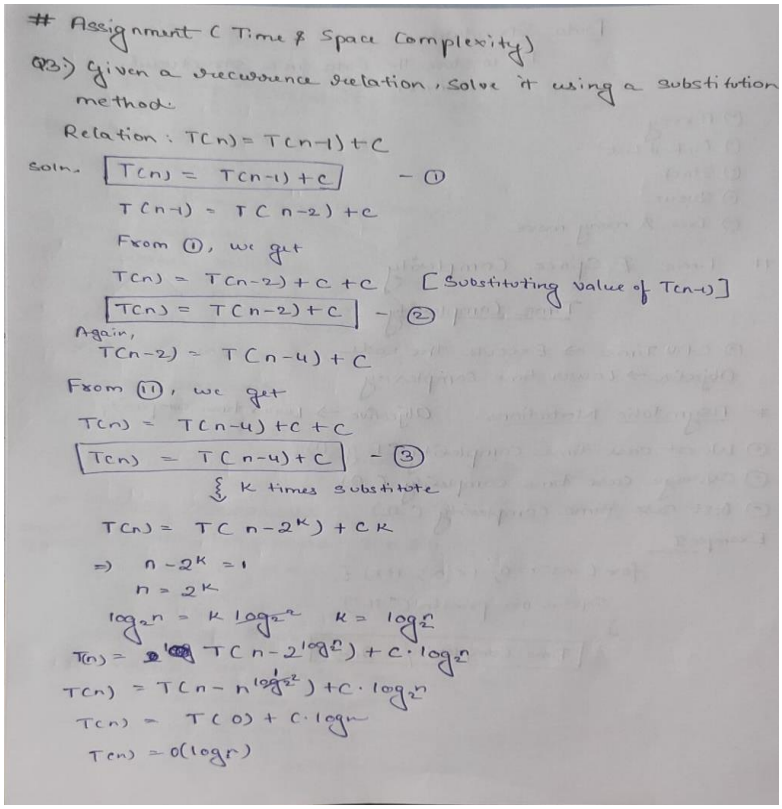
T(2)=3*27+24 [T(1)=27]

T(2)=105

Q3. Given a recurrence relation, solve it using a substitution method.

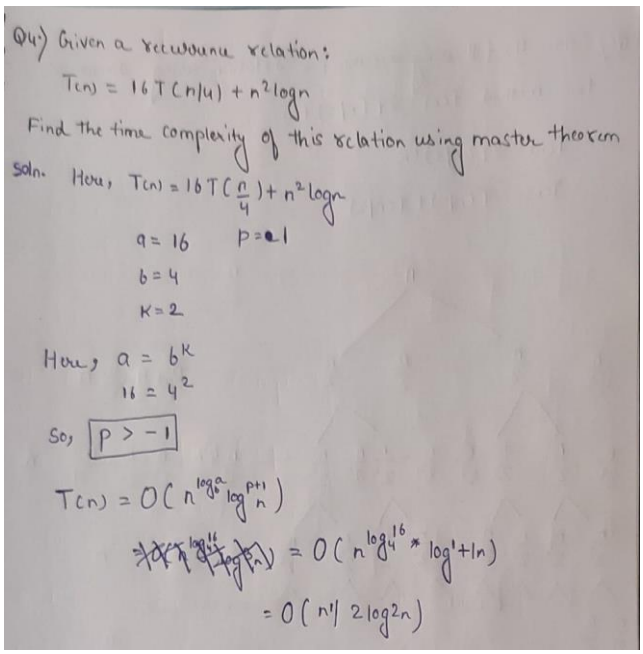
Relation : T(n) = T(n - 1) + c

Ans-



Q4.Given a recurrence relation: T(n) = 16T(n/4) + n2logn Find the time complexity of this relation using the master theorem.

Ans-

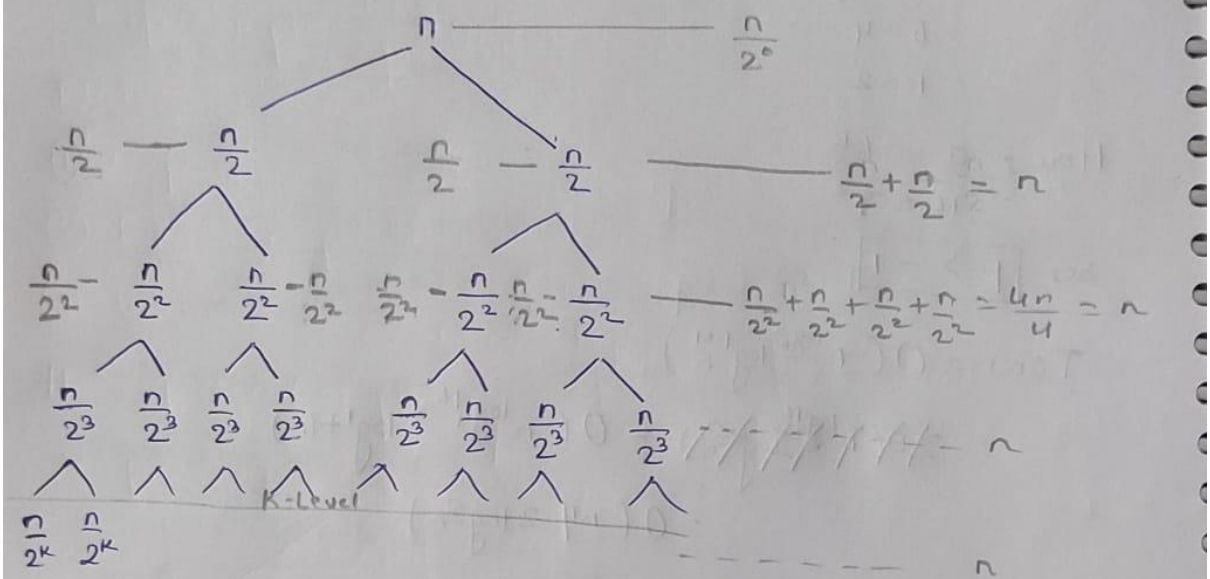


Q5.

Q5.) Solve the following recurrence relation using recursion tree method $T(n) = 2T(n/2) + n$.

Soln- $T(n) = 2T(n/2) + n$

$T(n) = T(n/2) + T(n/2)$



Time Complexity

$\left(\frac{n}{2^0} + \frac{n}{2^1} + \frac{n}{2^2} + \frac{n}{2^3} + \ldots\right)$ K-levels

$\frac{n}{2^K} = 1$

$n = 2^K$

$\log n = K \log 2$

$K = \log_2 n$

$n\left(\underbrace{\frac{1}{2^0} + \frac{1}{2^1} + \frac{1}{2^2} + \ldots}_{\log_2 n \text{ times}}\right)$ - G.P series

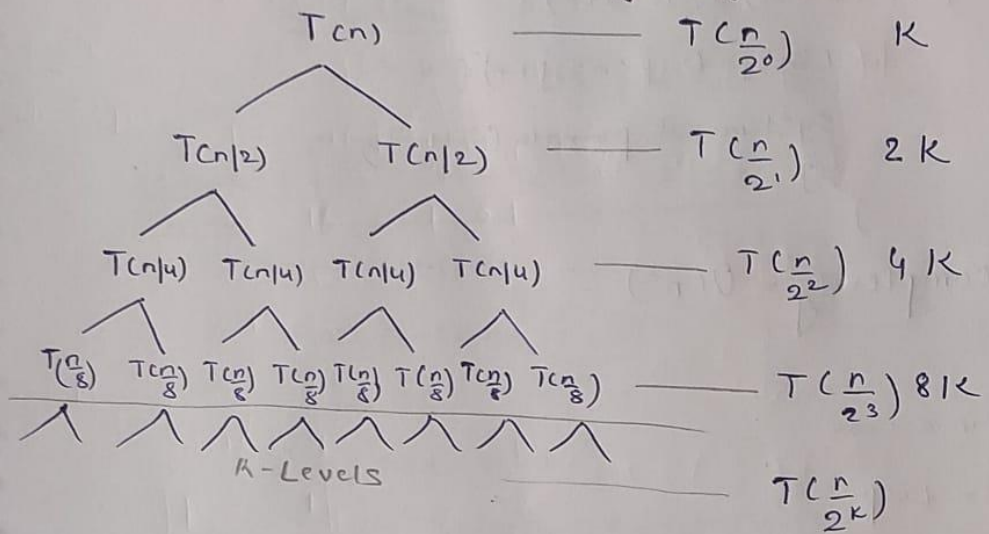
Sum of G.P series

$= \frac{a}{1-r} = \frac{1}{1-1/2} = 2$

Time Complexity = $O(n \log_2 n)$

Q6.

Q6) $T(n) = 2T(n/2) + K$, Solving using Recurrence tree Method



Time Complexity

$$\frac{n}{2^K} = 1$$

$$n = 2^K$$

Taking log both side

$$\log n = \log 2^K$$

$$\log n = K \log 2$$

$$K = \log(n) / \log(2)$$

$$K = \log_2 n$$

Cost at each level

$T(n) = K + 2*K + 4*K + \ldots + \log n$ times $+ O(1) * n$

$T(n) = K(1 + 2 + 4 + \ldots + \log n$ times$) + O(n)$

$T(n) = K(2^0 + 2^1 + 2^2 + \ldots + \log n$ times$ + O(n)$

G.P series,

$a = 1, r = 2$

$T(n) = K * (1/(2-1)) + O(n)$

$T(n) = K + O(n)$

$T(n) = O(n)$