Time and space complexity

Question 1. 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 is $O(n^2)$ as it uses nested loops. This is sum of first n natural numbers, this can be improved with the following code statement i.e n(n+1)/2
So that the time complexity will be O(n) know.

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

ANS. Given T(n) = 3T(n-1) + 12n and T(0) = 5

Substituting the values in the relation:
   T(1) = 3T(0) + 12
=> T(1) = (3*5) + 12
=> T(1) = 15 + 12 = 27

   T(2) = 3T(1) + 12 * 2
=>T(2) = (3 * 27) + 24 = 81 + 24
   Hence T(2) = 105.

Question 3: Given a recurrence relation, solve it using a substitution method.

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

ANS. Let the solution be T(n) = O(n), now let's prove this using the induction method.

For that to happen T(n) <= cn where c is some constant.
T(n) = T(n - 1) + c
T(n - 1) = T(n - 2) + c
T(n - 2) = T(n - 3) + c
|
|
T(2) = T(1) + c
—------------------------ Adding all above equations
T(n) = T(1) + cn
Let us assume T(1) to be a constant value.
T(n) = k + cn
Therefore, T(n) <= cn
Hence we can conclude T(n) = O(n).

Question 4: Given a recurrence relation:

T(n) = 16T(n/4) + n2logn

Find the time complexity of this relation using the master theorem.

ANS. From the given recurrence relation we can obtain the value of different parameters such as a, b, p, and k.
The relation: T(n) = 16T(n/4) + n 2log n
Here, a=16, b=4, k=2, p=1
bk = 42 = 16
Here a=bk
Also p>-1
Hence T(n)=θ(n log ab*log p+1n)
Therefore T(n)=θ(n log 164*log1+1n)=θ(n1/2log2n)

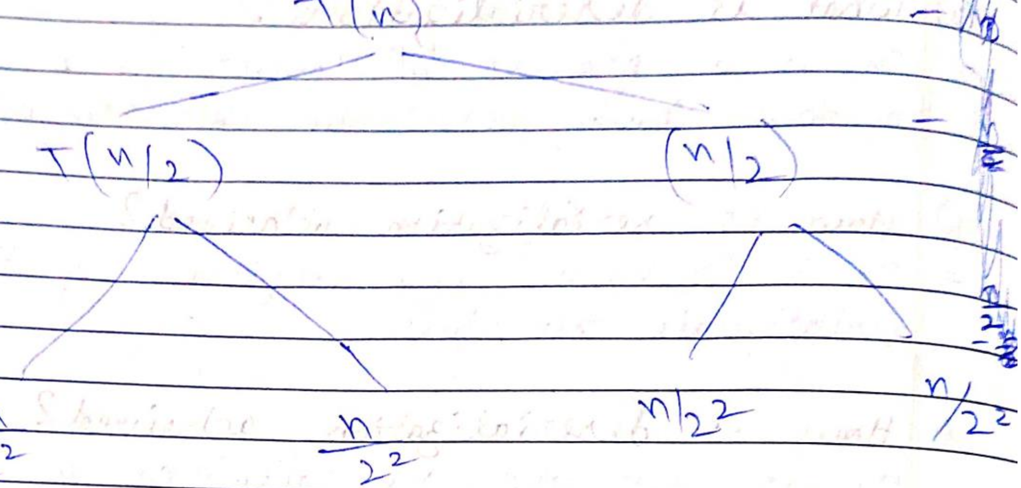
Question 5: Solve the following recurrence relation using recursion tree method T(n) = 2T(n/2) + n

Time and Space Complexity.

Q6) Solve $T(n) = 2T(n/2) + n$ using recursion tree?

Sol

Given $\Rightarrow T(n) = 2T(n/2) + n$



at k^{th} level

$k = \frac{n}{2^k}$

$\Rightarrow n = 2^k$

$\rightarrow$ taking log n both side

$\log_2 n = \log_2 2^k$

$\Rightarrow \log_2 n = k \log_2 2$.

$\Rightarrow k = \cancel{\log} n$ ~~total cost = log n.~~

$\therefore O(n \log n)$

DATE
PAGE No.

Total Cost = $k + 2k + 3k + 4k$

Total Cost $= k + 2k + 4k + \log(n) + O(1)^k$
$= k(1+2+4+ \ldots \log n) + n$

GP $\Rightarrow a = 1, r = 2 \Rightarrow a\left(\frac{r^n - 1}{r-1}\right)$

$\Rightarrow k \qquad \Rightarrow 1\left(\frac{2^k - 1}{2-1}\right)$

$\Rightarrow 2^k - 1$

$\Rightarrow k(2^0 + 2^1 + 2^2 + 2^3 \ldots \log n) + O(n)$
$\Rightarrow k + O(n)$

$\boxed{\therefore O(n)}$

Total cost $= n + n + n + n \cdot + k$

$= \cancel{k(n)}$

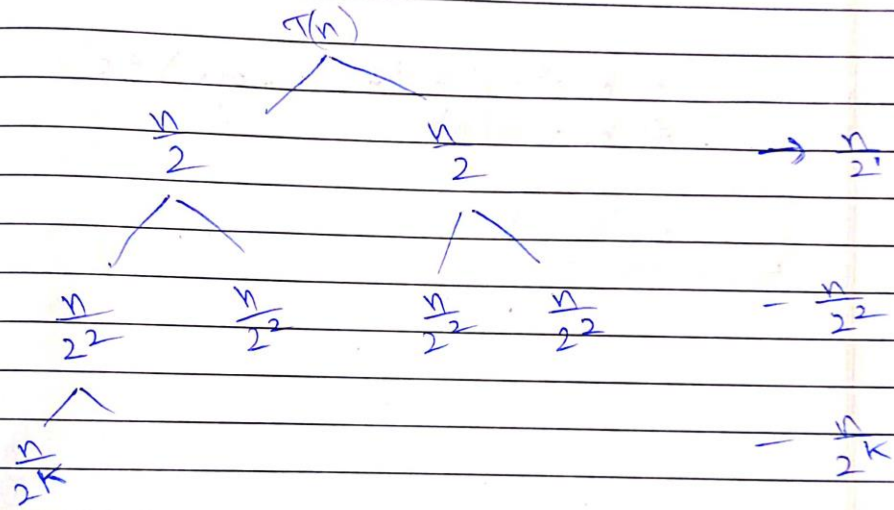
Total height of tree $= k$.

$\therefore k n = n \log n$

$\therefore O(n \log n)$.

Q6) $T(n) = 2T(n/2) + k$, solve using recurrence tree?

Sol



→ $\frac{n}{2^k} = 1$.

$n = 2^k$

$\log n = \log 2^k$

$\log n = k \log_2 2$

$\boxed{k = \log n}$

~~Total Cost = k + 2k + 3k + 4k~~

Total Cost $= k + 2k + 4k + \log(n) + O(1)^* n$

$= k(1 + 2 + 4 + \ldots \log n) + n$

GP $\Rightarrow a = 1, r = 2 \Rightarrow a\frac{(r^n - 1)}{r - 1}$

$\Rightarrow 1\left(\frac{2^k - 1}{2 - 1}\right)$

$\Rightarrow 2^k - 1$

$\Rightarrow k(2^0 + 2^1 + 2^2 + 2^3 \ldots \log n) + O(n)$

$\Rightarrow k + O(n)$

$\boxed{\therefore O(n}$